(continued from part 29)

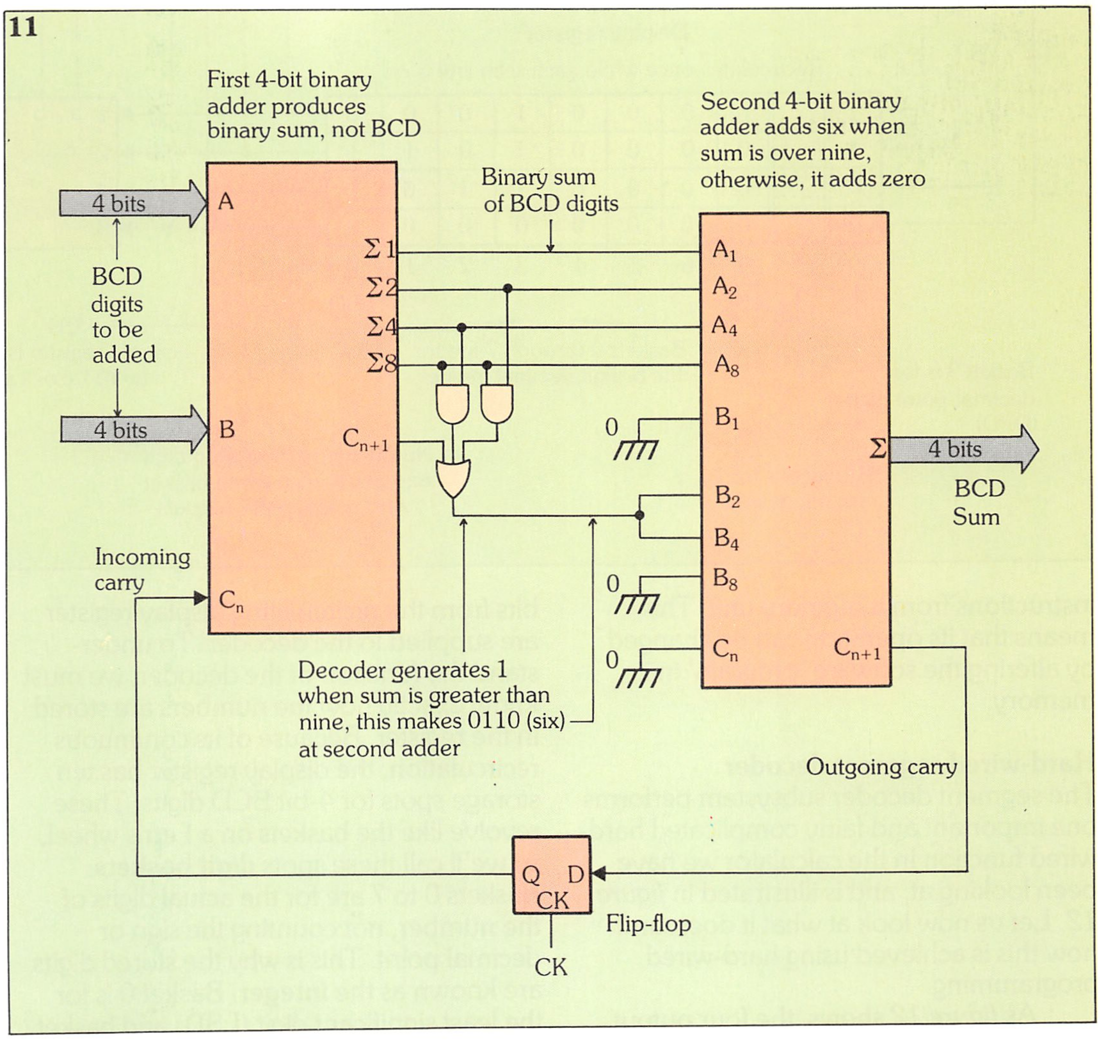
Figure 11 illustrates how this method of BCD addition could be used in a calculator. The adder-subtracter would have two 4-bit binary adders as shown, and a decoding network that emits a 1 when the sum is greater than 9, and a 0 otherwise. In the second adder, the binary sum is added to a number consisting of 0 in the ones and eights places, and the decoder output in the twos and fours places. So this number is six (0110) when the decoder recognises a number greater than nine. The output of the second adder gives us the 4-bit sum, while the carry bit is stored in the flip-flop.

In the examples of addition that we have just looked at, we have seen that bits can be grouped together for processing in different ways. The question of how the bits should be grouped arises in the design of a system's various stages of processing, and the final decision depends on careful consideration of how fast the function must be performed and how much processing capacity can be afforded.

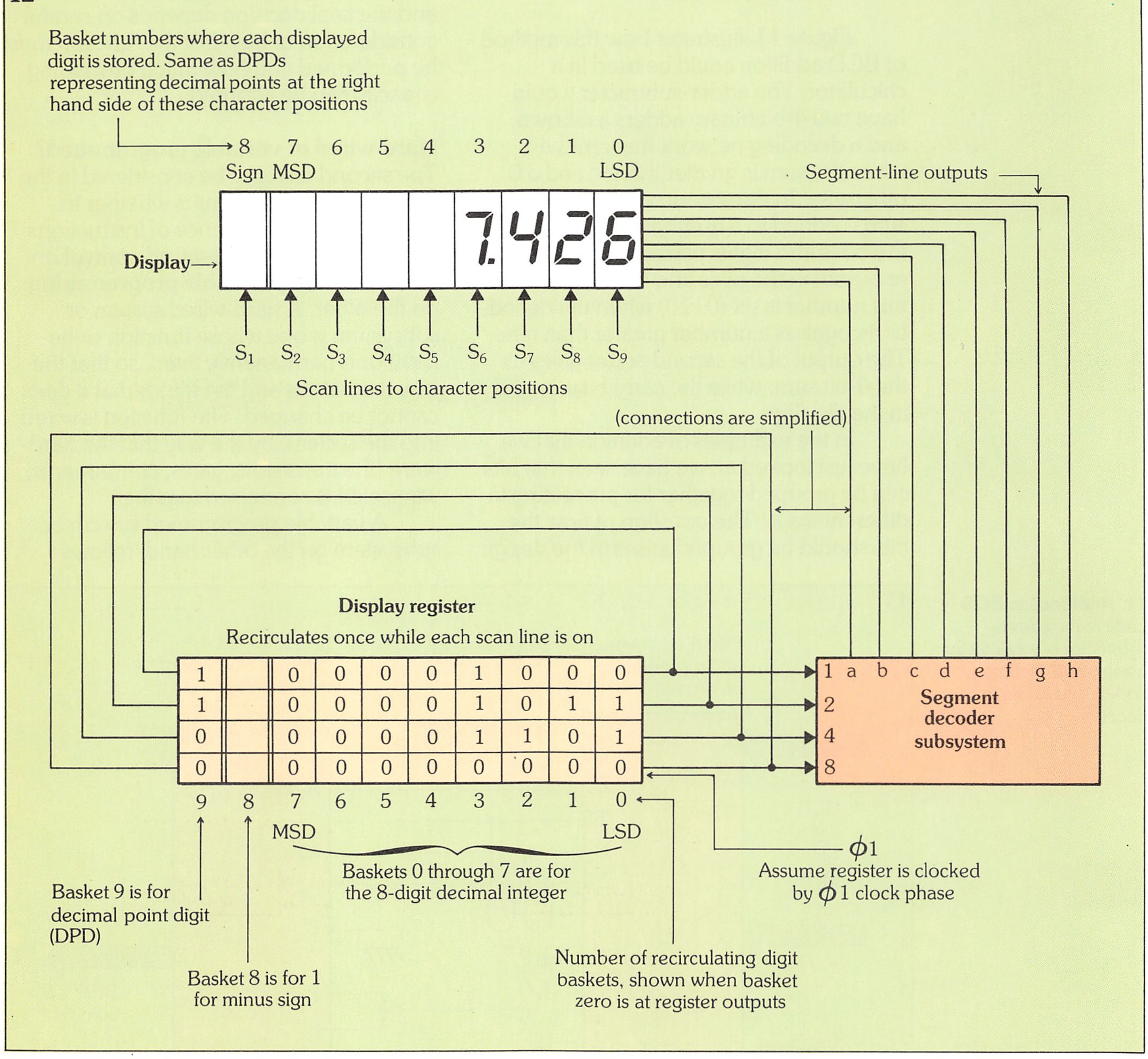
Hard-wired or variable programmed?
The second factor to be considered in the design of digital systems is whether to perform a given sequence of instructions by using **fixed** or **hard-wired control** on one hand, or by **variable programming** on the other. A hard-wired system or subsystem is one whose function or behaviour is permanently fixed, so that the way it operates and the things that it does cannot be changed. The function is wired into the system, by the way that the hardware (the transistors, gates, components, wires etc.) is connected together.

A variable programmed system or subsystem on the other hand, follows

11. Additions in BCD code in the adder-subtractor of a calculator using two 4-bit binary adders and a simple decoder.



12



instructions from a memory unit. This means that its operation can be changed by altering the software 'program' in its memory.

Hard-wired segment decoder

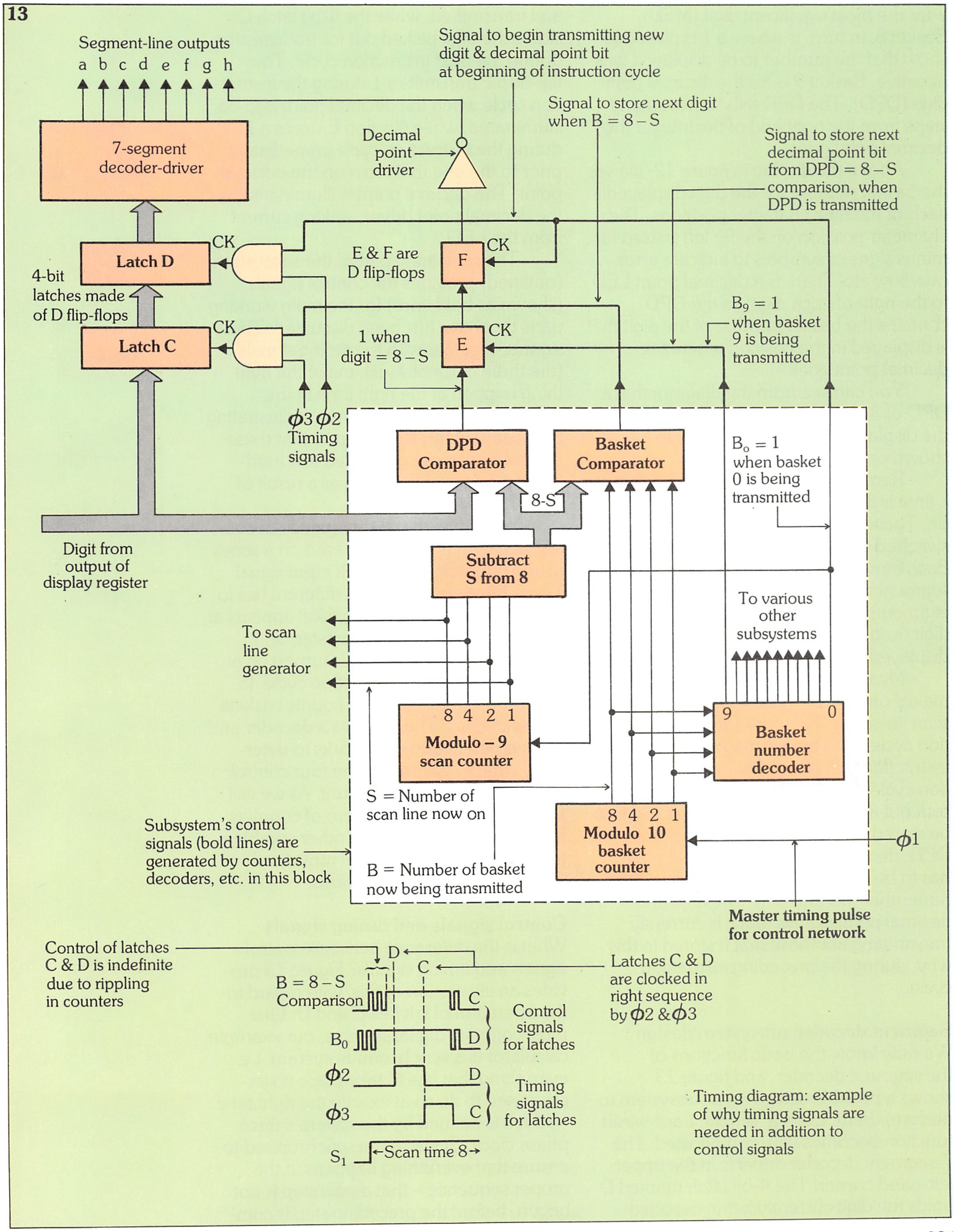
The segment decoder subsystem performs one important and fairly complicated hard-wired function in the calculator we have been looking at, and is illustrated in *figure 12*. Let us now look at what it does, and how this is achieved using hard-wired programming.

As *figure 12* shows, the four output bits from the recirculating display register are supplied to the decoder. To understand the function of the decoder, we must know exactly how the numbers are stored in the register. Because of its continuous recirculation, the display register has ten storage spots for 4-bit BCD digits. These revolve like the baskets on a Ferris wheel, so we'll call these spots **digit baskets**. Baskets 0 to 7 are for the actual digits of the number, not counting the sign or decimal point. This is why the stored digits are known as the **integer**. Basket 0 is for the least significant digit (LSD) and basket

12. The segment decoder subsystem selects and decodes the correct integer digit for the scan line that is 'on'.

13. A simplified design for a calculator segment decoder subsystem.

13



Timing diagram: example of why timing signals are needed in addition to control signals

7 for the most significant digit (MSD). Basket 8, in turn, is where a 1 is placed to show that the number to be displayed was negative. Basket 9 is for the decimal point digit (DPD). The DPD tells us how many steps from the right end of the integer the decimal point goes.

Above the display in *figure 12*, we see the basket number for the digit displayed in each of the nine character positions. The character position on the far left is used for minus signs or symbols to indicate error, overflow etc. There is a decimal point LED to the right of each digit, so the DPD contains the basket number of the digit that is displayed in the position where the decimal point goes.

You can see from the diagram that a DPD of 3 and an integer of 00007426 in the display register causes 7.426 to be shown on the display.

Remember, one character position at a time is illuminated when its scan line is on. These scan lines (S_1 to S_9) are switched on in sequence from left to right. Each time a new scan line is turned on, the segment decoder subsystem switches its eight outputs to the pattern for the next digit (and maybe decimal point) to be displayed in this next character position.

Meanwhile, the display register makes one full recirculation, while each scan line is on. This comprises one instruction cycle – the time during which one instruction is in effect. During each instruction cycle, the decoder subsystem has to pick out and store the integer digit that is to be stored next. It also has to check the DPD, decide whether the decimal point has to be turned on in the next place, and remember this decision. The digit and decimal point signals that it is currently transmitting are those that it stored in this way, during the preceding instruction cycle.

Segment decoder subsystem design

We now know the basic functions of the segment decoder, and *figure 13* shows a possible design for a subsystem to perform them. Even a 'simple' hard-wired function becomes very complicated. The 7-segment decoder driver is in the upper left-hand corner. The 4-bit latch marked D holds the digit currently being decoded and transmitted, while the 4-bit latch C stores the digit picked out for transmission during the next instruction cycle. The flip-flop F transmits a 1 during the instruction cycle when the decimal point is to be illuminated, while flip-flop E stores a 1 during the instruction cycle immediately prior to the one that turns on the decimal point. The decimal point is illuminated by the decimal point driver, sinking current from the LED.

The remaining part of the subsystem (outlined) provides the control signals (shown as bold lines) for the main working parts that have just been discussed. These signals, together with the timing signals (the three clock phases), make the right thing happen at the right time in the subsystem. The point that we're illustrating by looking at this subsystem is that these control signals are generated in a hard-wired manner, rather than as a result of stored instructions.

You'll find that the sequence of control signals in *figure 13* is based on a series of two counters and a clock input signal. This is because something different has to happen whenever a new basket appears at the output of the display register, and whenever a new scan line is turned on at the beginning of an instruction cycle. In effect, the control circuitry counts baskets and scan lines. Then, it uses a decoder and other combinational networks to determine what to transmit in the four control lines at each step in the count. As we will see later, this general pattern of counters followed by decoders and other combinational networks is typical of most hard-wired systems and subsystems.

Control signals and timing signals

What is the difference between control signals and timing signals? *Figure 13* provides an illustration of this, with regard to the operation of latches C and D. Like most complex digital systems, our example calculator is a **synchronous system**, i.e. everything that has to take place is synchronised to do so at exactly the right time – which is defined by the system's three phase clock. Systems are synchronised to ensure that everything happens in the proper sequence – that a new step is not begun, before the preceding step is com-

pleted. In effect, control signals say what to do, and timing signals indicate precisely when to do it.

In *figure 13*, the clock signal to each latch is the AND function of a control signal and a timing signal. As the timing diagram shows, the beginning of each control pulse is rather indefinite – due to the rippling in the counters. This could be a problem at the beginning of 'scan line 8', as scan line number eight comes on, because both latches are to be clocked during 'basket zero' time, but latch D must be clocked first.

This problem can be solved by making the clocking of the latches dependent on the clock phases as well as the control signals, ensuring that latch D gets clocked first. The clock signal says 'close the latch' and the timing signal says 'do it now!'

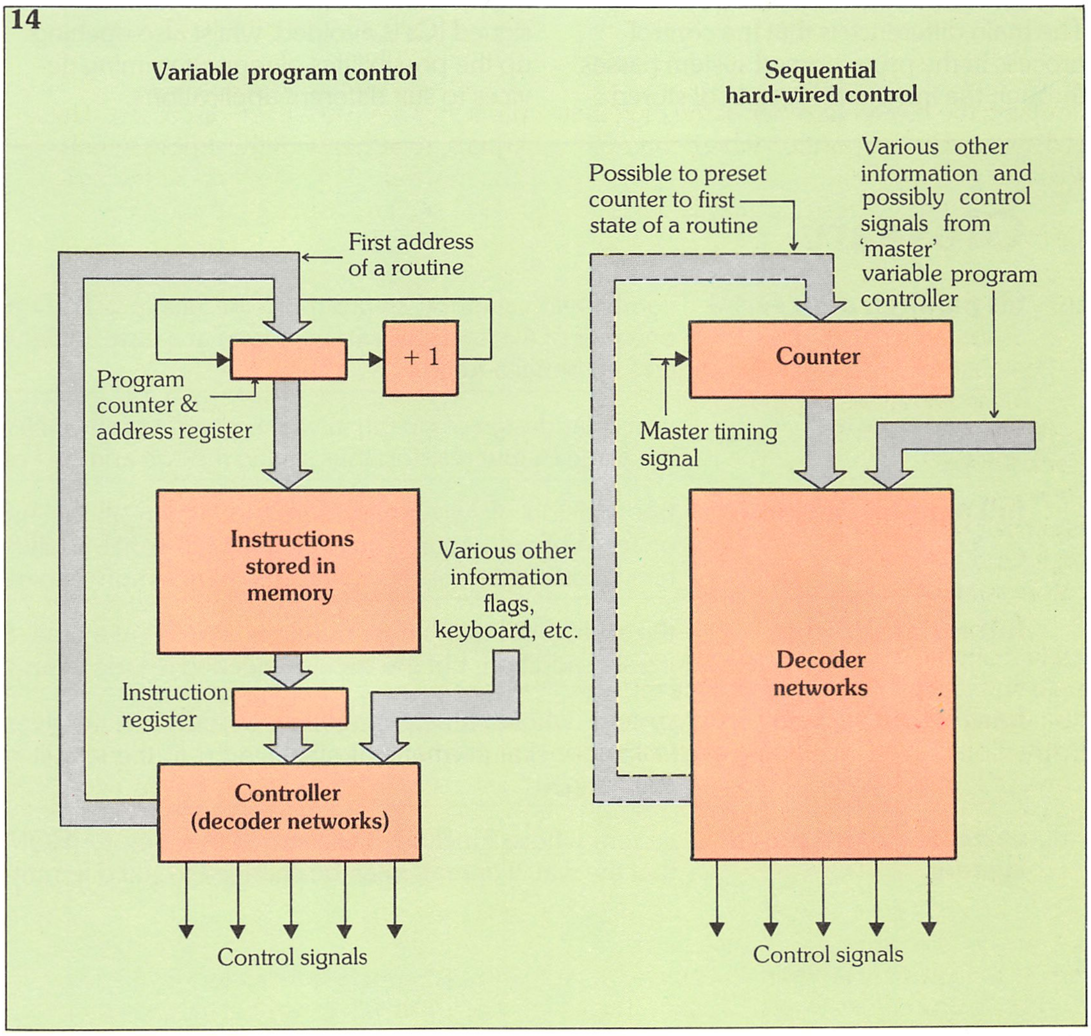
We now know that in a hard-wired system, different things have to be made to happen at different times in a pre-arranged sequence, based on counting clock pulses and decoding the counts. We also know about the partnership that exists between control and timing signals in a synchronous system.

Variable program control

The other option that can be used to control a sequence of operations is **variable programming**. We've already seen it in operation in the calculator example, because the 'master control' over all the various (hard-wired) sections of the calculator is exerted by following instructions stored in the microprogram memory (ROM).

Figure 14 shows a simplified general comparison of variable program and hard-wired control. You can see that the dia-

14. Simplified general comparison between variable program control and hard-wired control.



gram on the left is just a rearranged version of some parts from our calculator block diagram. The controller loads the first address of an instruction sequence (a routine) into the address register. The program counter steps through the routine in the same way as the hard-wired counter, until the controller interrupts by stopping the counting or loading of a new address. Each address eventually gives rise to a certain set of control signals, as does each counter state in the hard-wired system. The main difference is that the control process in the programmed system passes through the intermediate state of stored instructions.

The only serious disadvantage of variable program control is that it is generally slower than hard-wired control, and this is mainly due to the time necessary to fetch instructions from memory. Speed requirements prevent the use of variable program methods in many cases.

However, variable programmed control is being increasingly used, as the versatility and low cost of microprocessors means that the expense of custom designed ICs is avoided, whilst also opening up the possibilities of reprogramming devices to suit different applications.

Glossary

bit parallel, digit serial	combination method of grouping the bits in a BCD number, so that one pair of 4-bit groups can be added at a time. In between full parallel and full serial addition
expander gate	particular design of circuit, in which several positive NAND gates share the same output resistor, thus saving the size and cost of an IC chip
full parallel addition	the addition of two binary numbers, when all the bits are added at once. While obviously quicker than full serial addition, the circuitry needed is much more complex, and hence more expensive
full serial addition	the addition of two binary numbers, one bit at a time. Slower than full parallel addition, but the circuitry needed is simpler and cheaper
hard-wired system	system whose function (ability to perform a given sequence of instructions) is permanently fixed, so that the way it operates cannot be changed
variable programmed system	system whose function(s) are stored in memory (ROM or PROM), so that the way it operates can be changed to suit different circumstances

4

COMPUTERS & SOCIETY

Computing and music

Introduction

One of the first people to introduce electronic synthesis into music was the avant garde composer, Stockhausen. To many, the discordant and atonal nature of his compositions was unacceptable: traditional orchestrations were blended with banks of oscillators and ring modulators producing a range of totally new sound textures. Much of Stockhausen's work with electronic sound creation paved the way for future developments in both electronic music and synthesiser technology.

In the early 1960s, electronic music became more popular, largely through the work of Walter Carlos. Throughout the late fifties and early sixties, Carlos had been exploring the musical possibilities of electronics through various sound realisations. Like Stockhausen, he was forced to use very crude equipment. The creation and

Below: sound mixing in a recording studio using an integrated audio and visual command centre.

The Research House/Solid State Logic Ltd

performance of even simple sounds was usually a gruelling and futile experiment – crescendo and diminuendo, for example, the two most basic forms of musical expression, required the most calculated and laborious manipulation of volume controls and filters.

Carlos then collaborated with Robert Moog, who was attempting to produce a commercially acceptable implementation of the synthesiser, and together they developed the subsystems which were essential if the synthesiser was to be converted from a machine which produced an electronic noise into something which could be creatively employed in making music. With this step, Carlos became the first true electronic musician.

More recently, the French composer/performer Jean Michel Jarre has set the direction for most electronic music. Jarre's compositions are based on many layered sequences which weave an intricate pattern throughout the music, and his use of sound textures and rhythmic patterns gives a good impression of an electronic symphony orchestra. He was one of the first composers to apply programmable computer sequencing to the creation of rhythm.

When performing live, the composer would be surrounded by banks of computer and electronic hardware. Performances were split between time spent at the keyboard overlaying live playing against the programmed backing, and shifting plugs on a patchboard to alter the sounds or sequence of the backing track. This 'wall of electronics' has now been replaced by commercially available synthesisers, which run under computer control.

Early electronic music, of course, was restricted by the level of the technology which was available. In many instances, the composers themselves prompted the design of more complex equipment. Now, musicians are confronted with a dazzling array of synthesisers, computers and rhythm machines, each with a huge potential for the creation of music in the electronic medium. For some of these instruments it might take many lifetimes to fully explore all of the available sounds, timbres and rhythms which the new computer technology has made available.

First applications

Computers were first applied in the musical field to the problem of **sequencing**. In a musical sense, a sequencer is a device which allows notes to be stored in some sort of memory in the order in which they were played, and then replayed later.

In most analogue synthesisers it is the frequency of oscillations of a voltage-controlled oscillator which produces a note of a certain pitch. One octave on the keyboard is equal to one volt; so each key creates a set voltage value depending on its position on the keyboard. The change in voltage as different keys are pressed produces changes in the frequency of oscillations, thereby changing the pitch of the note.

To build a sequencer these voltage changes need to be fed into and stored by a computer. Analogue-to-digital converters are required to convert the analogue voltage values into digital values which can be stored in memory. To play back the stored sequence, digital-to-analogue converters reconvert the digital values into analogue voltages and feed them into the synthesiser (*figure 1*).

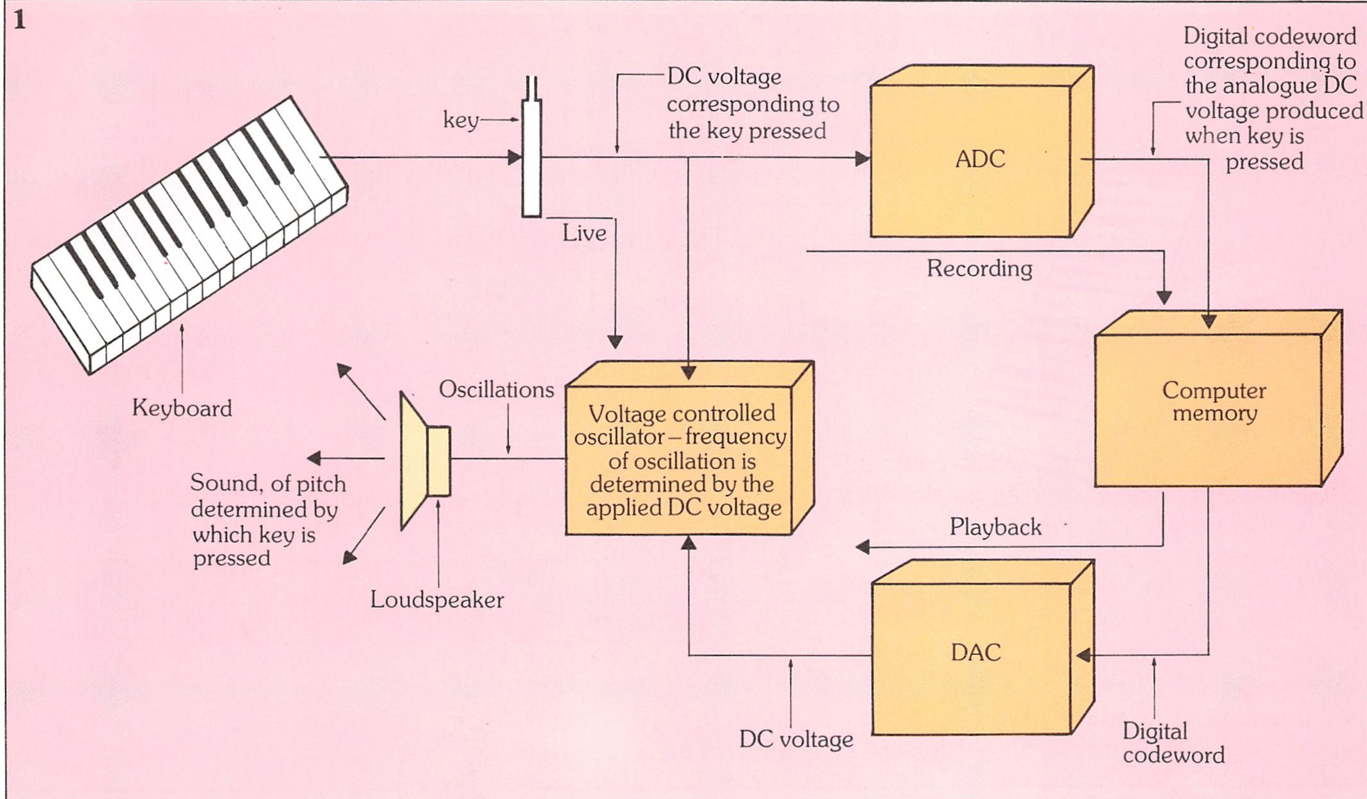
Modern computer sequencers are capable of more than a simple record and replay function – the ability to program the sequencer in both **real** and **step time** has been one of the most useful recent developments. Early sequencers were only programmable in step time. Step timing divides the computer memory into a series of units, in each of which can be stored the information relating to one **event**; in each of these events a note could either occur or not occur. It makes no difference whether the note is recorded towards the beginning or the end of the event, the note still appears exactly on the beat on replay.

Music programmed into a step time sequencer, therefore, has a 'robotic' sound. This staccato sound can be smoothed by using more events of a shorter duration and by allowing notes to be tied between two or more events. However, this system does have its drawbacks. For instance, to transcribe a piece of music into the sequencer, it must first be broken down into its smallest note value, and longer notes or rests must be built from multiples of the event time. (*Figure 2*).

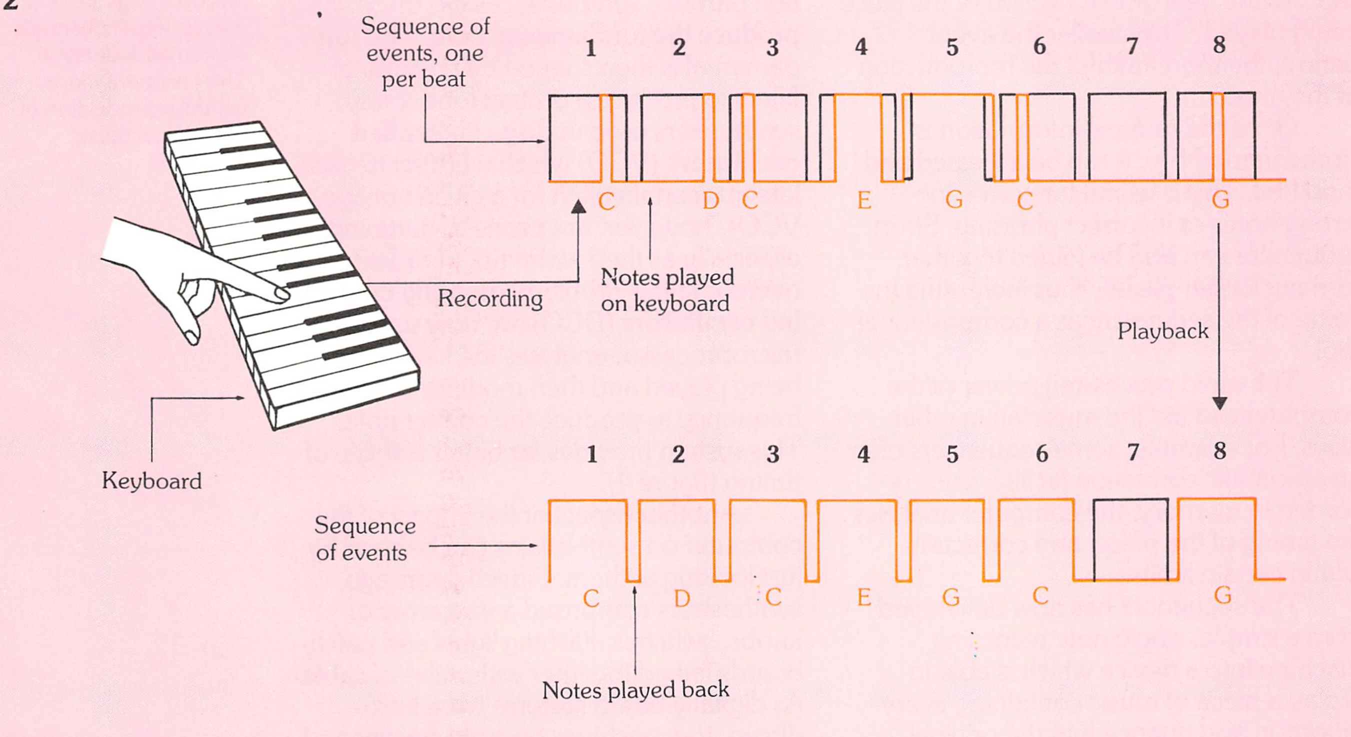
1. A simplified diagram illustrating the operation of a sequencer.

2. During step time recording it makes no difference whether the note is recorded towards the beginning or end of an event – it is still replayed exactly on the beat.

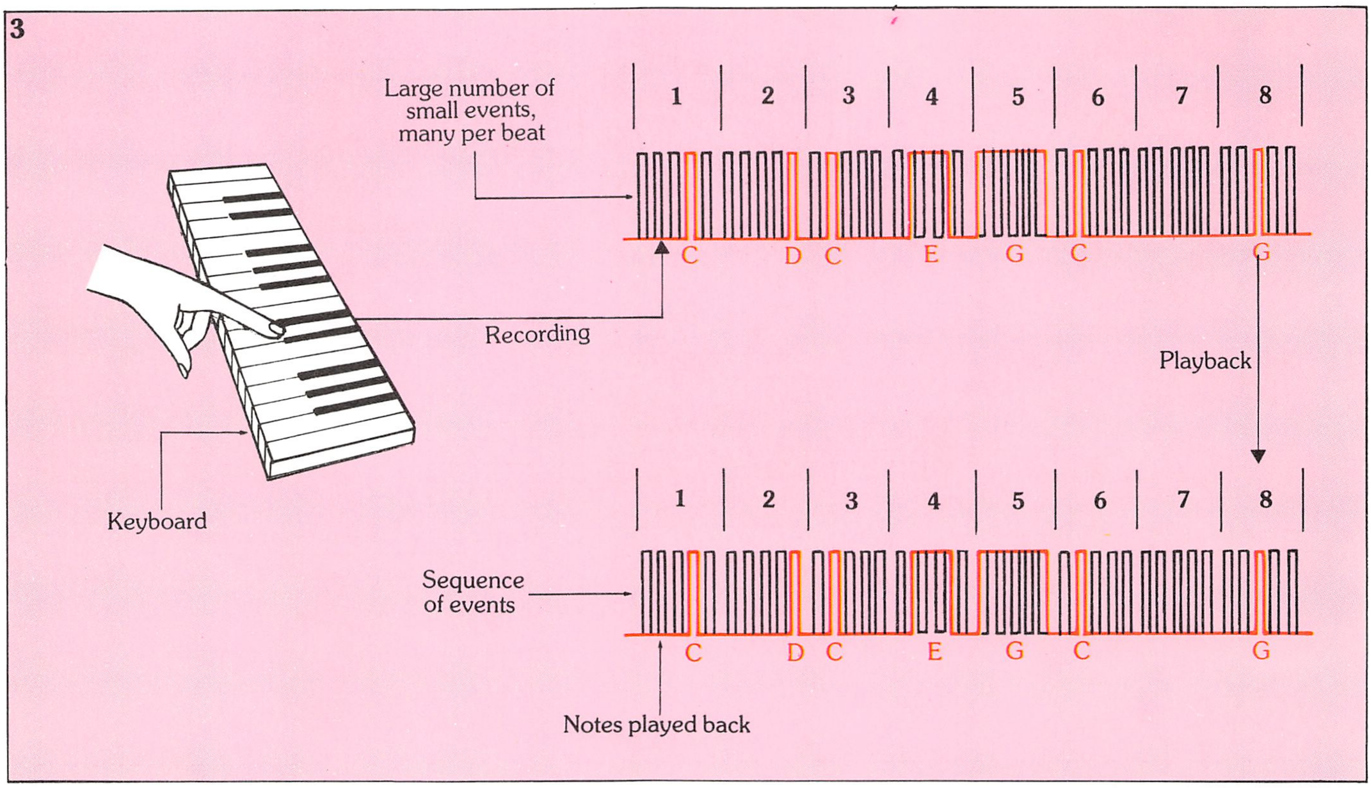
1



2



The computer of a real time sequencer, on the other hand, sets up a large series of events, each covering a very small event time. Music is then programmed in by playing it in real time, usually against a metronome pulse provided by the computer (*figure 3*). The short time periods over which the keyboard is sampled for note



3. A real time sequencer sets up a large series of events, each covering a very small event time. This ensures a more faithful reproduction of the original music.

information, means that it is easier to capture the 'feel' and inflection of the piece being played. The smaller the event period, the more faithful the reproduction of the music.

Once the musical information is stored in memory, it can be accessed and modified. This is useful for correcting wrong notes or incorrect phrasing. Short sequences can also be joined together, forming longer pieces, thus increasing the value of the sequencer as a compositional tool.

The rapid processing power of the computer can aid the musician in other ways. For example, some sequencers offer an automatic correction facility: once recorded in memory, the computer analyses the timing of the piece, and corrects it within certain limits.

The sequencer has now developed from a simple, single note recording machine into a device which is able to replay a piece of music containing every inflection and nuance that the original contained.

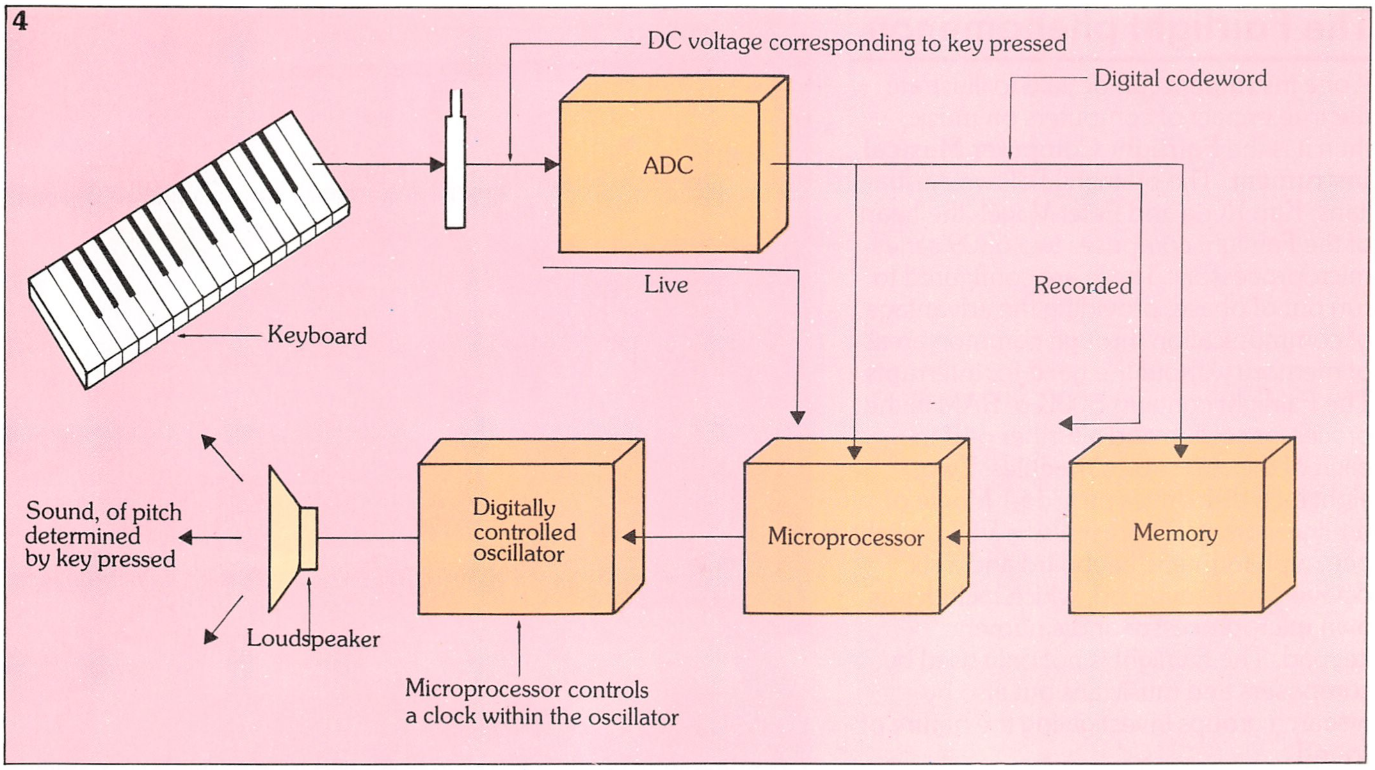
Computers in synthesisers

While early synthesisers were electronic, today's machines are microprocessor controlled. The oscillators are the most important part of a synthesiser as it is these which produce the fundamental note. This fundamental is then shaped by an array of filters to produce a distinct tone. Early synthesisers used **voltage controlled oscillators** (VCO) which were set to oscillate at a certain pitch for a given voltage. VCOs, however, are prone to detuning, especially as they warm up. In order to overcome this problem, **digitally controlled oscillators** (DCO) are now used. A microprocessor analyses the key that is being played and then modifies a clock frequency to produce the correct note. This system provides far better stability of tuning (*figure* 4).

Another aspect of the impact of the computer on synthesisers can be seen by just looking at them. Fifteen years ago, synthesisers comprised a vast array of knobs, switches, flashing lights and patchboards linked together with miles of cable. As digitally based systems were introduced, the machines became smaller and tidier, but a number of knobs, sliders and switches were still required to recall a sound.

Preset synthesisers represented the first step in reducing the number of con-

4



The Research House/Solid State Logic Ltd

4. The use of digitally controlled oscillators overcomes the problem of detuning which occurs when voltage controlled oscillators warm up.

Above: setting up the studio for full audio/visual recording.

trols. These had a set of preprogrammed patches stored in ROM which could be instantly transferred to the operating section of the synthesiser, enabling changes between sounds to be made at the push of a button.

The next step was to allow users to program their own sounds directly into RAM memory – this meant leaving the knobs and sliders intact but providing a way in which their values could be copied and stored. This system provided additional scope for performing musicians who could develop sounds at their leisure and write them into memory, to be recalled instantly, live. This information is written into CMOS RAM which can be kept active by a small internal battery (so that the information is not lost when the power is switched off). For extra security, many synthesisers offer a cassette tape or floppy disk back-up system enabling the information to be written to a standard tape recorder in much the same way as a home computer stores its programs and data.

Whilst this system is very useful, there are still a large number of physical controls. With this problem in mind, Robert Moog designed the **Moog Source** synthesiser, the only sound editing controls being a set of touch sensitive buttons, a large rotary potentiometer and an LED display. To edit or create a sound, all that is necessary is to select which control needs modifying by pressing the appropriate button – its value then appears on the LED display showing that it is ready for editing. The value of the parameter is altered by rotating the potentiometer and the new value is then rewritten into memory. Many of the later model synthesisers have opted for this system for reasons of cost and ease of use.

The Fairlight phenomenon

If one instrument can be said to illustrate the true impact of computers on music, then it is the **Fairlight Computer Musical Instrument**. The brainchild of two Australians, Kim Ryrie and Peter Vogel, the heart of the Fairlight comprises two 6809 series microprocessors. These are configured to run out of phase, providing the advantage of communication through common areas of memory without the need for interrupts. The Fairlight contains 500K of RAM in the processor section and a further 64K in each of the eight voice modules. Two, eight inch disk drives provide 1 Mbyte of on-line storage. There is also a VDU, a light pen, alphanumeric keyboard and a six octave piano keyboard, which includes its own microprocessor and a numeric keypad. The Fairlight is not only used by composers and musicians but also by research groups investigating the nature of sound.

The basis of the Fairlight's power lies in its ability to sample and modify natural sounds. Utilising a system of very fast analogue-to-digital conversions, the Fairlight is capable of reading and storing a sound from either a microphone or a direct line input. Once the sound is stored, it can then be reproduced in the correct pitch over the entire length of the keyboard. For more accurate reproduction, the sound can be sampled several times for different positions on the keyboard, thus enabling the software to capture the changing character of the sound as its pitch increases or decreases. Of course, if different sounds are sampled for different parts of the keyboard, then a whole orchestra could be played simultaneously. The different sounds do not need to be split, they can all be arranged to trigger off a single note.

When using the Fairlight, sounds do not need to emanate from the real world, they can be created via the light pen and VDU. The user only needs to draw the waveform of the sound on the screen, and then play it back from the keyboard. The software can also calculate waveforms, so both the initial and final waveform can be drawn, with the computer being left to fill in the gaps inbetween. Using this process, it is possible to use natural sounds in an unnatural way, for example, the note may begin sounding rather like a grand piano and as it dies away it may take on the characteristics of a bowed cello.

Fairlight software also contains a very powerful compositional language. A piece of music can be scored by writing basic notation as a program. The completed **piece** can contain up to eight **parts**, which in turn can be made up of a maximum of thirty **sequences**. Once the piece has been written, it can then be edited and copied to disk.

The great advantage of the Fairlight is that it provides the composer with total control over the finished work. Until now, composers have had to rely on the interpretations of other people to produce the finished piece. Using this technology, composers can write, record, and play back their compositions as if they were being played on the actual instruments.

Above: the Fairlight Computer Musical Instrument. (Photo: CEM Italia).

Computers in control

One of the most interesting developments in the use of computers in music over the last few years has been the arrival of **MIDI** (Musical Instrument Digital Interface). The basis of MIDI is a system by which computers, sequencers, drum machines and synthesisers can all be linked together. Two of the leading synthesiser manufacturers, 'Sequential Circuits' and 'Roland', drew up the basic specification for MIDI in January 1983 and it is now being used in most new synthesisers.

MIDI enables the transfer of information regarding a note and how it is played from one instrument to another. This means that a musician can play an electric piano, say, whilst at the same time control a synthesiser and a drum machine. The benefits for live performance are considerable, but the benefits for the musician are even greater.

There are already a number of commercially available interfaces available for connecting MIDI equipment to ordinary computers. The computer can be used in a variety of ways, for example as a simple real or step time sequencer. The musician would play the piece on a MIDI equipped synthesiser, and the computer would store the incoming signals from the MIDI channel in memory, to be replayed at a later date. Because the MIDI system allows a number of MIDI devices to be linked together, several sequences could be stored using one synthesiser and then replayed simultaneously using several synthesisers — one for each part. To aid the writing of simultaneous sequences, the computer can be made to replay earlier sequences for the musician to listen to when overdubbing the next part.

Below: the Emulator 'Digital Synthesiser Base'.

ZEFA/Syco Systems

Taking the sequencer idea a stage further, it is possible to work on the stored sequence through the computer. This way, errors can be corrected and the score modified using either the computer's keyboard, or indeed the synthesiser's. Some of the software that is now becoming available enables the user to write directly into the computer, in a similar way to the Fairlight's music language. With a suitable graphics package it is possible to see a score on the screen whilst listening to the computer playing it back on the synthesiser. Some packages also include a 'dump to printer' facility. Conversely, notes can be written onto staves on the screen using a keyboard or light pen and then played back via the synthesiser.

Information on sound dynamics can be sent through the MIDI channels, and so for the first time, a computer is able to record the velocity with which the keys are being struck and to reproduce these on playback. Program parameters can also be passed through MIDI — sound changes can be stored in the computer's memory, and replayed during the piece.

Already it is becoming unnecessary for the synthesisers to be attached to a keyboard as musicians can work entirely through a compositional language on the computer, playing it back through a MIDI equipped unit containing simply the necessary oscillators and filters to create the sound. Companies like 'Roland' are producing stand alone piano and polysynthesiser modules to be used in conjunction with either a computer or another MIDI synthesiser as controller. For the home-based computer musician this means that professional quality sounds are becoming available in an acceptable price bracket.

Computers for music

When choosing a home computer for musical work a decision has to be made as to whether the computer is to produce the sound, or if it is to be used as a controlling unit for another piece of sound production hardware. Some computers have very good internal sound production capabilities, whilst others are easy to use as controllers; some are capable of both. Software is, of course, a vital element also. To start, we'll look at some computers which produce sounds from their own internal sources.

Yamaha CX5

The CX5 conforms to the new MSX specification and anyone familiar with Yamaha's work in the synthesiser field will be aware of the new ground that they have been breaking with their FM synthesis system. It is this part of the computer which is of most interest to us.

The CX5 has what amounts to the core of Yamaha's DX9 synthesiser built in to it. The computer is configured as being **eight note polyphonic**, which means that eight notes can be played and heard simultaneously. There is room for 96 different sounds: 48 of them as presets and a further 48 accessable via additional software. The CX5 has a real time sequencer with sufficient memory for about eight minutes of music, and the machine also has a rhythm unit. There is a three and a half octave mini-keyboard supplied; full-size keyboards can be added. A MIDI interface box which can either control or be controlled by other MIDI equipped instruments can also be added.

The music software is mainly available on ROM cartridges; the two most interesting packages being the *Voicing Program* and the *Music Composition Program*. The Voicing Program enables the parameters of each different sound to be called up on the screen and edited. This has the considerable advantage over normal synthesisers in that the high resolution screen conveys far more information about what the parameter is actually doing. Another feature of this program is that it allows the sound being edited to be placed on one half of the keyboard while the original sound is retained on the other half so that direct comparisons can be made. Once the sound is complete it is then written into memory or stored on a data cassette.

The Music Composition Program takes a more formal approach to the process of writing music. The screen display comprises a stave displayed in two parts: music is input onto the stave by a combination of the music keyboard for pitch, and the typewriter keyboard for note information. (The typewriter keyboard can be used for both purposes if desired.)

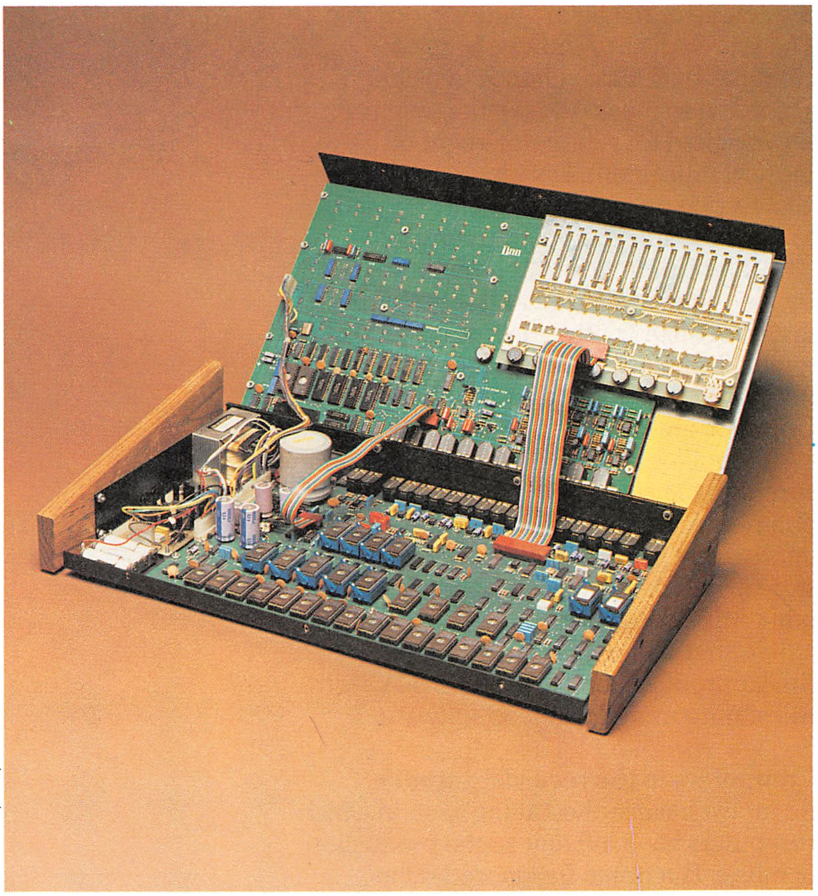
Up to eight parts can be written in, and musical expressions such as crescendo and pianissimo can be added later. Once the composition is finished, it is then either stored or replayed using the CX5's internal sounds or via MIDI. One problem is that unless there are eight different MIDI sounds available, they will all be played back in one sound.

Anyone listening to popular music now cannot help but notice that most of the sounds present are not of natural origin: strings, brass and drums are all being produced synthetically. The instruments which produce these sounds owe more to computer technology than to any musical heritage. Musicians now have to be as familiar with an alphanumeric keyboard as with a piano keyboard.

Apple II

Despite the disadvantage of not possessing an internal sound generator, the Apple II is becoming increasingly popular. Its accessable memory expansion buss makes it easy to add circuits onto the main board, thus making it a natural choice to house sound generation hardware.

The Alpha Syntauri system from the Syntauri Corporation was one of the first commercial computer based music systems available and it uses the Apple II as the host computer. The heart of the Syntauri system is a bank of 16 digital oscillators which are paired to give eight note polyphony. Control over these oscillators is provided by the software which is supplied with the system. Its programmable features are two ADSR (attack, decay, sustain, release) type envelopes per sound, one envelope per waveform (see *Digital*



ZEFA/Syco Systems

Above: interior of the Linndrum digital drum machine.

Electronics 24). There is preset storage for up to 20 banks of presets, and up to 200 distinct instruments can be stored per disk.

Vibrato can be added to any sound, with both the rate and the depth being software controlled. Transposition can be implemented in quarter tone steps over ten octaves and the software can scan for and implement velocity sensitivity over the entire range. Various pre-programmed effects, such as chorus, tremolo and pitch bend, can be patched onto the keyboard and provisions are made for users to create and patch in their own effects.

One of the most impressive new developments is the **Metatrak** package which provides a sixteen track digital recording facility, resident in the Apple. For the composer, the Syntauri with Metatrak becomes a highly flexible scratchpad for trying out ideas and refining pieces prior to recording. Also, with this software, the performing musician has a computer capable of providing a complete orchestral backdrop to any live work.

Using Metatrak enables individual tracks to be mixed and remixed without any degradation of quality. This has the advantage that a musician can complete a piece before entering a conventional recording studio where every minute costs money. The amount of note storage is only limited by the Apple's memory: with the Metatrak expander, between 30 and 60 minutes of music can be stored.

The **Mainframe Enterprise**, a plug-in board developed in response to a particular band's desire to upgrade its rhythm sequencer (developed on their Apple) is a new development. The band wanted to replace the analogue drum sounds with digitally stored 'real' drum sounds. The Mainframe Enterprise actually goes one step further as the plug-in card is able to sample and replay any sound!

With the accompanying software to edit and store the sounds, the band now has an Apple which can sample real sounds in the same way as the Fairlight CMI – hardware restrictions, however, mean that the sample quality is not quite as good as the Fairlight, but it is certainly comparable to more expensive machines. It is hoped that the band will shortly bring out a version to run with the Commodore 64.

BBC Micro

For those people more interested in teaching music via computer, the BBC Micro is a natural choice as it is the standard computer for the teaching establishment.

Internally it is capable of producing three note polyphony, with comprehensive, if difficult to understand, envelope control. One of the major problems with using the BBC Micro as a recording or performing device is that the sound output is routed to a small, low quality internal loudspeaker. In order to transfer the output to a usable device, it is necessary to open the case and solder directly onto the tracks on the PCB. Not something which most users would like to chance.

Despite this shortcoming, there is a considerable amount of educational software being developed; the themes centring around the teaching of music theory rather than performance skills.

There is already a MIDI interface

available, complete with sequencing and editing software.

Commodore Business Machines
What makes the Commodore 64 particularly useful is the inclusion of a sound interface chip: SID. This chip contains a three voice polyphonic synthesiser which is programmable by the user. The ADSR can be set for each of the three voices. Four different wave shapes can be selected for each voice: square, pulse, sawtooth and white noise. Limited filtering is possible, as is variation of pulse width.

The sound quality of the output is not as good as a normal synthesiser, but it can be improved by feeding it out to a separate amplifier. With a little experiment, some quite impressive sounds can be produced.

The disadvantage with the Commodore 64 is the difficulty in accessing the sound registers. Most computers provide commands in their BASIC for directly accessing the sound generator, however, with this machine the user is left to work out which part of the sound chip is to be modified and then resort to altering the stored values individually. There are now quite a few packages available which make this task easier.

Two programs which help the user take advantage of the Commodore 64 sound facilities are *Synthy 64* and *Dancing Feet.* The Synthy 64 package forms a subset of the BASIC language. Changes to the sound registers are written in using a simple identifier to indicate to the program which register is to be changed. The real power of Synthy 64 lies in its use as a compositional language. Standard musical notation is used to write notes, rests, pitch and timing information; sequences can be overlayed to produce chords or counter melodies. The resulting 'score' can be stored on disk or tape as a BASIC program. Because of the speed of execution, sound parameter changes can be made in the middle of a piece with no discernible delay.

The Dancing Feet program allows the musician to attempt to play a totally new instrument. All the user input is fed into the program using an ordinary games joystick. The direction in which the joystick is moved is translated into a change in either the octave or the pitch of the melody line. The screen display shows a series of coloured bars which change in height according to the volume of the note they are assigned to. Whilst not a serious musical instrument, Dancing Feet does allow a person with no musical knowledge to create simple melody structures.

The Commodore computer was used by 'Sequential Circuits' as the basis for their first MIDI control interface. The 'Sequential Circuits' 64 sequencer plugs into the cartridge port on the back of the Commodore 64. The operating software for it is built-in on ROM. Basically, the cartridge transforms the 64 into a powerful polyphonic sequencer. It will accept MIDI information from any MIDI device and store it in real time, playing it back on any other MIDI device.

Other companies are developing similar products for use with the 64 which will feature more advanced software.

Computers in the recording studio
So far we have looked at the ways in which computers have influenced the production of music. But not to be ignored is the influence of the computer in the recording studio. Perhaps the most impact the computer has made has been through the digital recording techniques it has made available.

One of the biggest problems in any recording situation is the build up of tape hiss. Many systems have been tried to overcome this, the most notable being the **Dolby** system. The problem with all of the noise reduction systems available is that however good they are, they always interfere in some way with the original signal; a digital system reproduces nothing but the original signal.

A digital system works in the same way as the Fairlight, but on a much bigger scale. The information being fed into it is converted into a digital equivalent and it is this which is stored on the tape. To recreate the sound the information is fed back into the encoding computer which interprets it and produces an analogue signal. Because the information is stored in digital form on the tape, it is not affected by tape hiss.

The other advantage of this system is

Above: the Solid State Logic Stereo Video System enables a single operator to control a comprehensive array of signal processing and routing capabilities – all available in a single unit. In addition to mono and stereo outputs for live teleproduction, this console provides: 24 group outputs for multi-track recording; a 6 group post-production mixing matrix; complete input/output and monitoring controls; high and low pass filters; 4 band parametric equalisation; an expander/gate; a compressor/limiter; and other features.

that overdubbing can be performed without any increase in background noise. Pitch changes can be carried out in the same way that the Fairlight shifts the pitch of its sampled sound. This does not have the effect of lengthening or shortening the recording to which the old method of changing the tape speed was prone.

One of the most interesting applications of a computer in a studio environment is that of the **computer controlled mixing desk**. One of the biggest problems for a producer when mixing down a recording is remembering what settings need to change at which points during the piece. Often during the course of a single song many changes have to be made to the levels of the instruments and their tone characteristics, not to mention any effects or signal processing which need attention.

By using a computer monitor and storing every change which is made to the desk, all changes can be built up step by step until the final mix is ready. So that the computer remains in time with the piece as it is being mixed, a timing pulse is recorded on a spare track and the computer is locked onto this.

To produce the final mix, the producer goes through the recording one track at a time, listening to and making any changes, which the computer records. Changes can be made later by entering an edit mode and modifying the computer's record of the mix. When the producer is finally happy, control can be handed over to the computer to carry out the final mix.

One added advantage of this system is that should anything happen to the master once it has left the studio, another identical mix can be created in a matter of seconds using the original multitrack recording and the changes stored in the computer.

ELECTRICAL TECHNOLOGY

Frequency response of networks

Almost all signals – whether they form music, speech or a television signal – can be made up of a number of sinusoids at different frequencies. It is obviously very important to know how different circuits respond to sinusoidal voltages over a wide range of frequencies. When these frequency response characteristics are plotted on a graph, it is convenient to use a scale where equal increments along the axis correspond to equal multiples of frequency. This type of scale is known as a **logarithmic scale**.

Pure inductors and capacitors

By now we should know that the magnitude of the impedance of a pure inductor increases linearly with frequency, from zero at zero frequency. The impedance of a pure capacitor, on the other hand, decreases with frequency, tending to zero at infinite values of frequency. The red curves in *figure 1* illustrate this effect for a 16 mH inductor and a 1.6 μF capacitor.

If we now consider the impedance of a real inductor that has an inductance of 16 mH and a resistance of 3 Ω, we obtain the curve shown in black. As you can see, at high frequencies the characteristic matches that of the pure inductor, while at low frequencies, it flattens out to a value of 3 Ω.

Figure 2 illustrates the phase characteristics of these circuit elements. As expected, the phase of the impedance of the capacitor is –90° at all frequencies, showing that the voltage lags the current by 90°. Similarly, the inductive impedance's phase is +90° at all frequencies. The real inductor (series combination of inductor and resistor) however, has a phase of +90° at very high frequencies, but at low frequencies the impedance approximates to a resistance, so the phase angle tends to be 0°.

Returning to *figure 1*, the graphs of the impedance of the pure capacitor and inductor both have slopes of exactly 45° when they are plotted with frequency and impedance on logarithmic scales. This is completely independent of these elements' magnitudes.

Looking at the real inductor (inductor and resistor in series), we can see that the graph changes from a 45° slope to a horizontal straight line at a frequency of about 30 Hz. The point at which this occurs is known as the **corner frequency** or the **break frequency**.

Here we see that the reactance of the pure inductor is 3 Ω, which is exactly the same as the magnitude of the resistance. This property is completely general for all circuits of this form; the impedance graph changes its slope at a frequency where the reactance is equal to the resistance. We can also see from *figure 2*, that the phase of the impedance is 45° at this corner frequency.

Let's take the case of an LCR series resonant circuit which we looked at in an earlier *Basic Theory Refresher*. The impedance graph for this network is shown in green in *figure 1*. At low frequencies, the impedance of the inductor is small, while that of the capacitor is large; so the graph approximates to that of the pure capacitor. At high frequencies, the impedance of the inductor becomes large and the graph approximates to that of the pure inductor. As you can see, at some intermediate frequency – about 1 kHz here – the reactance of the capacitor cancels that of the inductor and the total impedance becomes equal to the value of the resistance, 3 Ω.

Figure 2 shows the phase characteristics of this circuit. At low frequencies, the phase is –90°, at high frequencies it is +90° and at the resonant frequency of 1 kHz, the phase angle is zero. We can also see that the phase changes from –90° to +90° over a frequency range of about 600 Hz to 1.6 kHz.

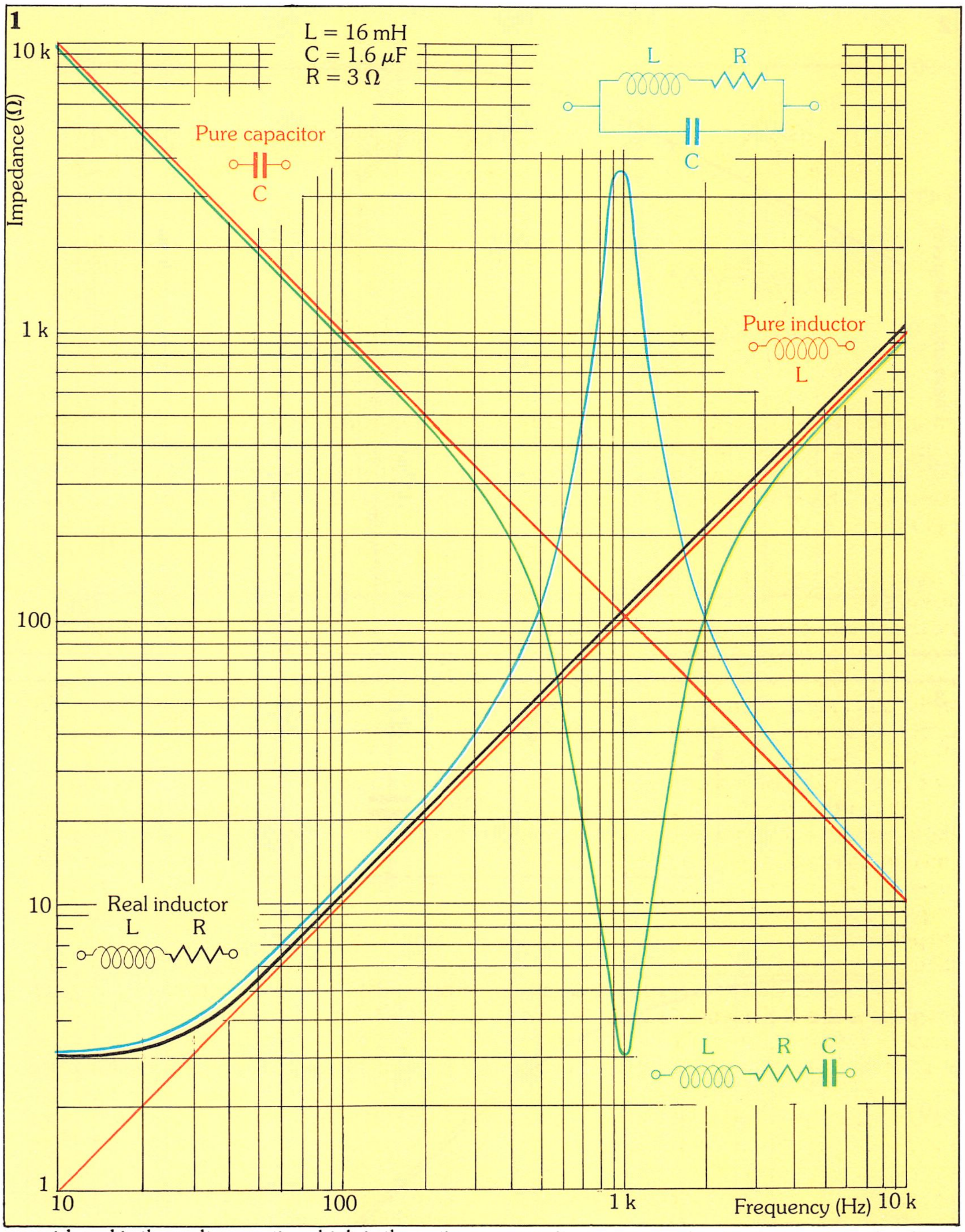
If we were to replot these curves for a circuit with a larger resistance – say 30 Ω – we would find that the change of phase from –90° to +90° was more gradual over a wider range of frequency; at the same time the magnitude curve (*figure 1*) would show a valley which was much broader. This fact is important in designing highly selective tuned circuits which have a very narrow bandwidth.

A similar curve for a parallel resonant circuit is shown in blue in *figure 1*. At low frequencies this approximates to the inductor-resistor combination, and at high frequencies to a pure capacitor.

Two-port networks

We now know how the impedance (ratio of V/I) of a network is related to frequency, and these ideas can be applied to the ratio of the input and output voltages or currents in a two-port network. The most usual ratio that is

1. Impedance *vs* frequency for a pure inductor and a pure capacitor.



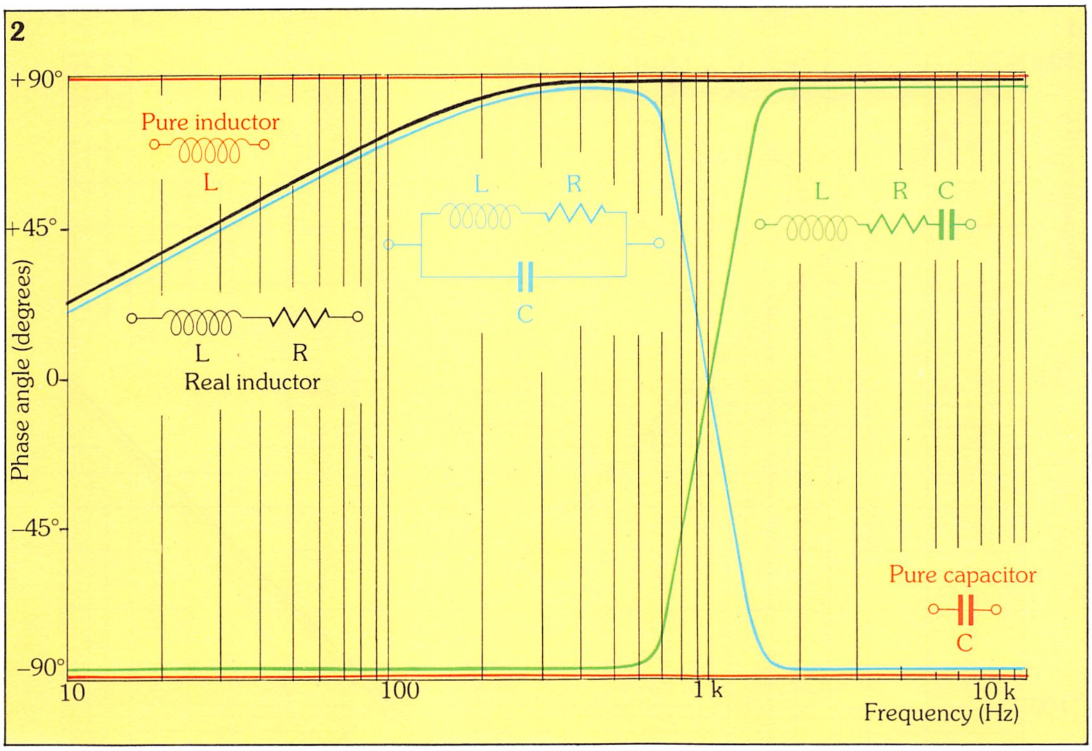
considered is the voltage ratio which is the ratio of output to input voltage. This generally consists of a magnitude and a phase, so graphs are needed for both. We have previously seen that it is convenient to measure the magnitude by its logarithm using units of decibels.

Frequency response of two-port networks

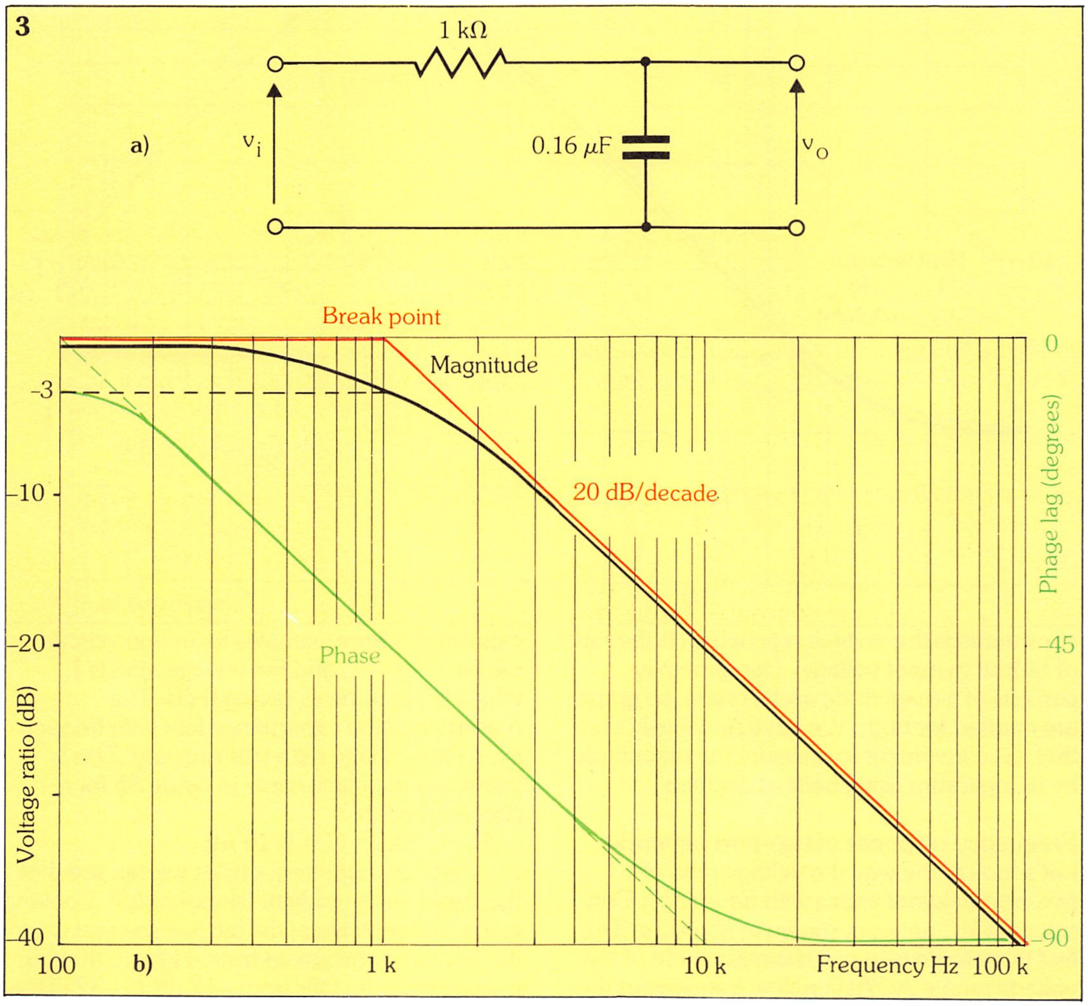
Let's look at the way the voltage ratio of a two-port network varies with frequency. Consider the RC network shown in *figure 3a*. The first thing to do is to plot the magnitude of the voltage ratio v_o/v_i. At very low frequencies the capacitance approximates to an open circuit, so we have $v_o = v_i$, and the voltage ratio is 1, which is the same as saying 0 dB. The reactance of the capacitance falls with frequency, so the voltage ratio falls similarly. This is shown by the black curve in *figure 3b* for a circuit in which:

$R = 1\ k\Omega$; $C = 0.16\ \mu F$

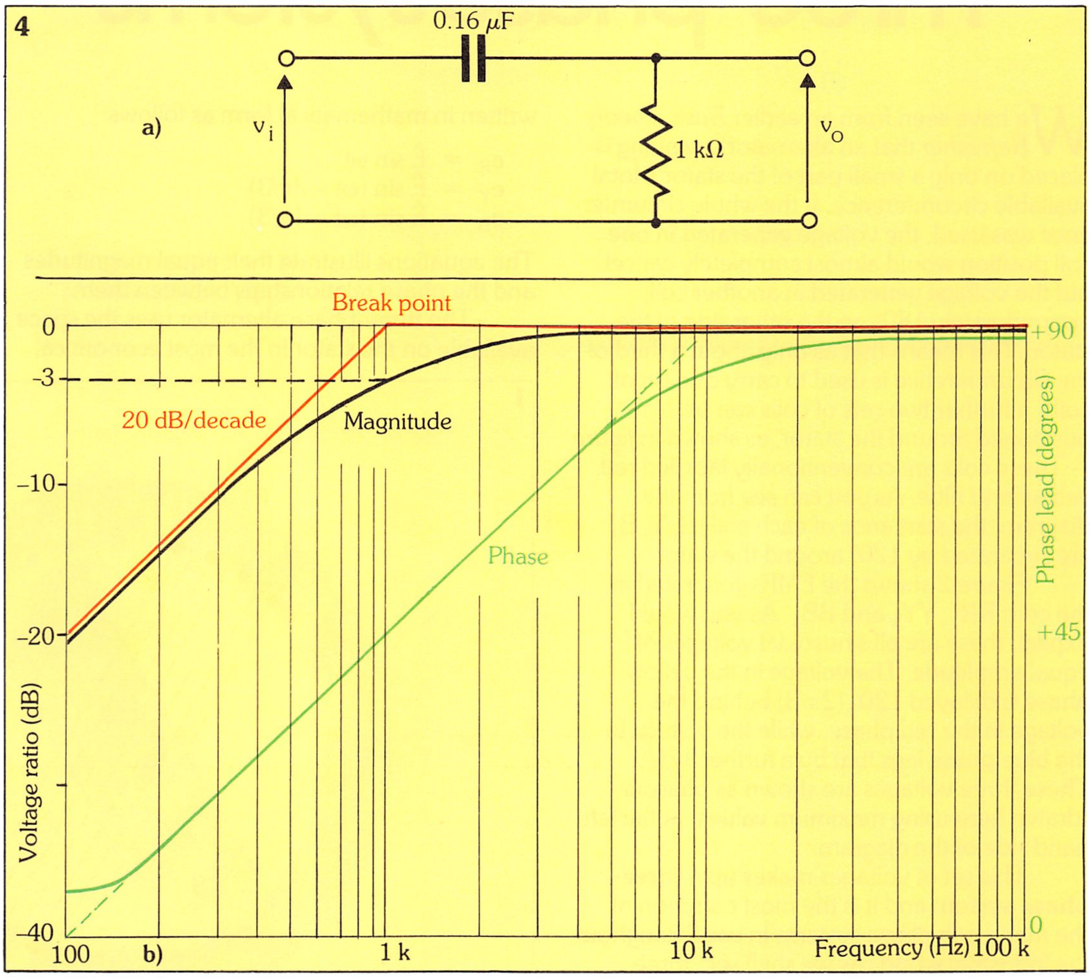
At very high frequencies we can see that the curve has a gradient (slope) of 45°. Looking at the decibel scale at the left, we see that when the frequency increases from 4 kHz to 40 kHz, the voltage ratio falls from –12 dB to –32 dB: a



2. Phase characteristics of a pure inductor and a pure capacitor.



3. (a) Two-port RC network; (b) voltage ratio *vs* frequency.



4. (a) Two-port RC network with the capacitance as the series element; **(b)** voltage ratio *vs* frequency.

fall of 20 dB. Similarly, when the frequency changes from 10 kHz to 100 kHz, the voltage ratio falls from –20 dB to –40 dB. In fact, anywhere along this high frequency part of the graph the voltage ratio falls by 20 dB when the frequency increases by 10. This voltage ratio **roll-off** is usually quoted as 20 decibels per decade (20 dB/decade) since a decade corresponds to a tenfold increase of frequency. This same roll-off can also be expressed as 6 dB/octave, an octave being a change of frequency by a factor of 2.

The corner frequency at which the voltage ratio changes from horizontal at 0 dB to a slope of 20 dB/decade occurs at a frequency of 1 kHz. This corresponds to the frequency at which the reactance of the capacitor is equal to 1 kΩ and is exactly equal to the value of the resistor.

It is often convenient to approximate this graph using the two straight lines shown in red on *figure 3b*. The first is a horizontal line at 0 dB up to the break frequency and above that a straight line falling with a slope of 20 dB/decade. Such approximations are known as **Bode plots** or **asymptotic plots**.

This network's phase characteristics are shown in green. At low frequencies, the phase shift between v_o and v_i is zero, but at very high frequencies the output voltage lags v_i by 90°. At the break frequency the phase shift is exactly –45°. This phase characteristic may also be approximated by straight lines (shown in *figure 3b* in broken green). The first is a line passing through –45° at the corner frequency, and continuing to 0° and –90° at frequencies of a decade below and a decade above. Below and above these frequencies, horizontal straight lines approximate the phase characteristic at 0° and –90°.

Consider the circuit in *figure 4a*. Here the capacitance is the series element and the resistor is connected across the output. The variation of gain with frequency is shown in black in *figure 4b*. The two straight red lines show the Bode plot which has a break point at 1 kHz, is horizontal at 0 dB above this frequency and falls with a slope of 20 dB/decade below 1 kHz. The phase characteristic has a similar shape to the one in *figure 3*. Here the phase shift is +90° at low frequencies, falling to zero at high frequencies and has a phase shift of 45° at 1 kHz, the break frequency. □

ELECTRICAL TECHNOLOGY

Three-phase systems

We have seen from an earlier *Basic Theory Refresher* that an alternator's winding is placed on only a small part of the stator's total available circumference. If the whole circumference was used, the voltage generated in one coil position would almost completely cancel out the voltage generated in another coil, approximately 180° on the other side of the stator. This means that as only about a third of the circumference is used to carry one set of coils, a further two sets of coils can be positioned around the stator, as shown in *figure 1*. These coils are conventionally labelled red, yellow and blue. As you can see from the diagram, the start ends of each coil (R, Y, B) are separated by 120° around the stator.

Figure 2 shows the EMFs generated in the coils RR_1, YY_1 and BB_1. As you would expect, these are all sinusoidal voltages of equal amplitude. The voltage in the yellow phase is delayed 120° ($2\pi/3$) behind the voltage in the red phase, while the voltage in the blue phase lags that by a further 120°. These three voltages are shown as phasors (drawn here using maximum values) at the left hand side of the diagram.

This set of voltages makes up a **three-phase system** and it is the most common of the many polyphase systems in use throughout the industrial world. As we shall see, three-phase systems enable considerable economies to be made in both generating and distribution equipment.

The three voltages, e_R, e_Y and e_B can be written in mathematical form as follows:

$$e_R = \hat{E}\sin\omega t$$
$$e_Y = \hat{E}\sin(\omega t - 2\pi/3)$$
$$e_B = \hat{E}\sin(\omega t - 4\pi/3)$$

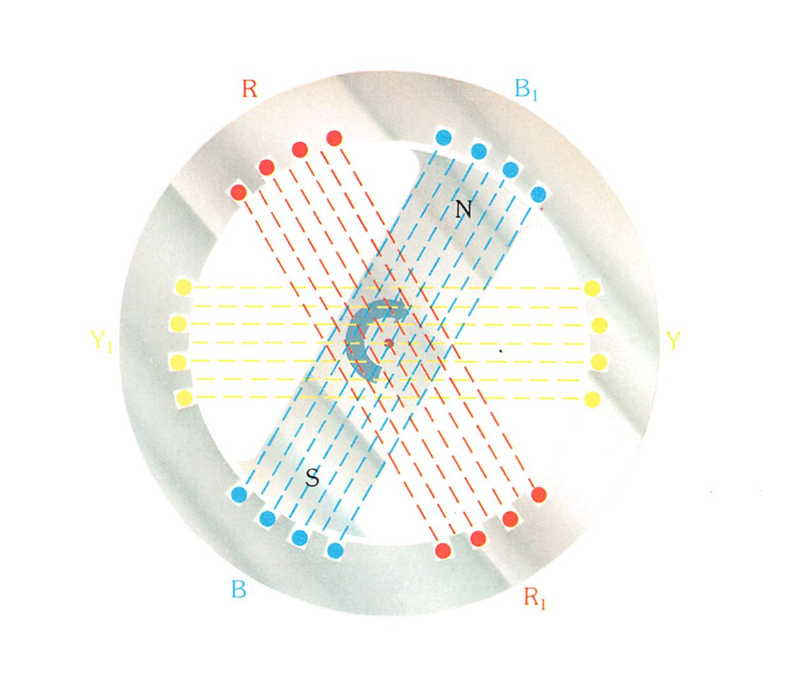
The equations illustrate their equal magnitudes and the phase relationships between them.

The three-phase alternator uses the space available on the stator in the most economical

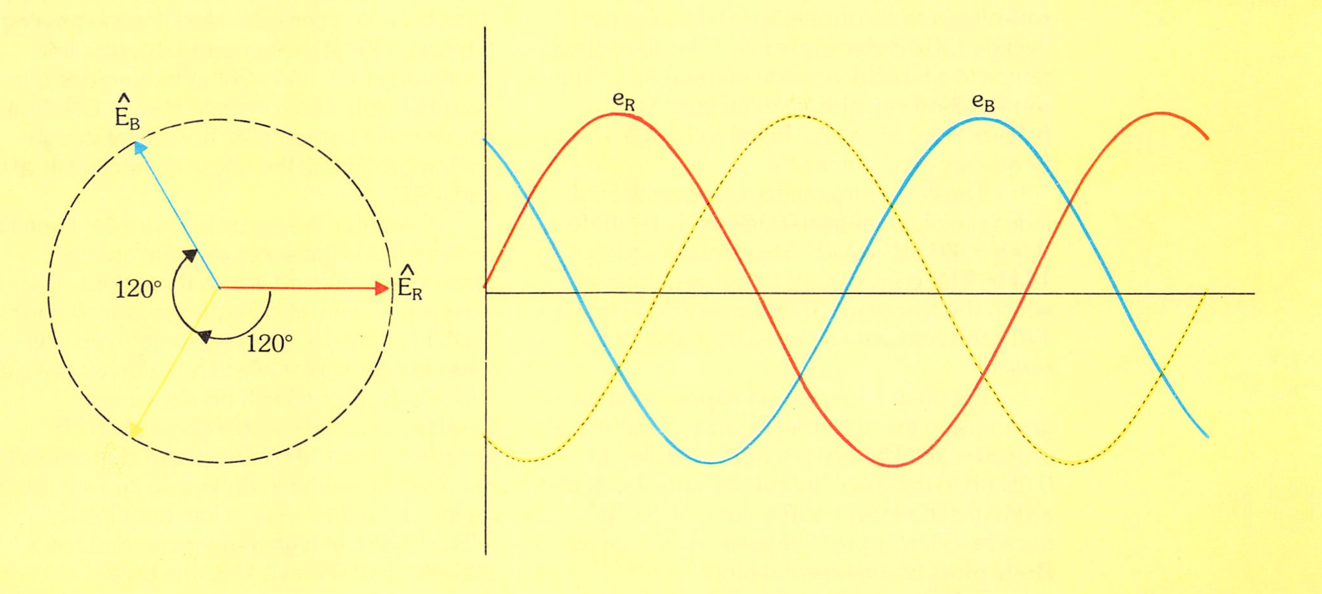
1. Three sets of coils positioned around the stator.

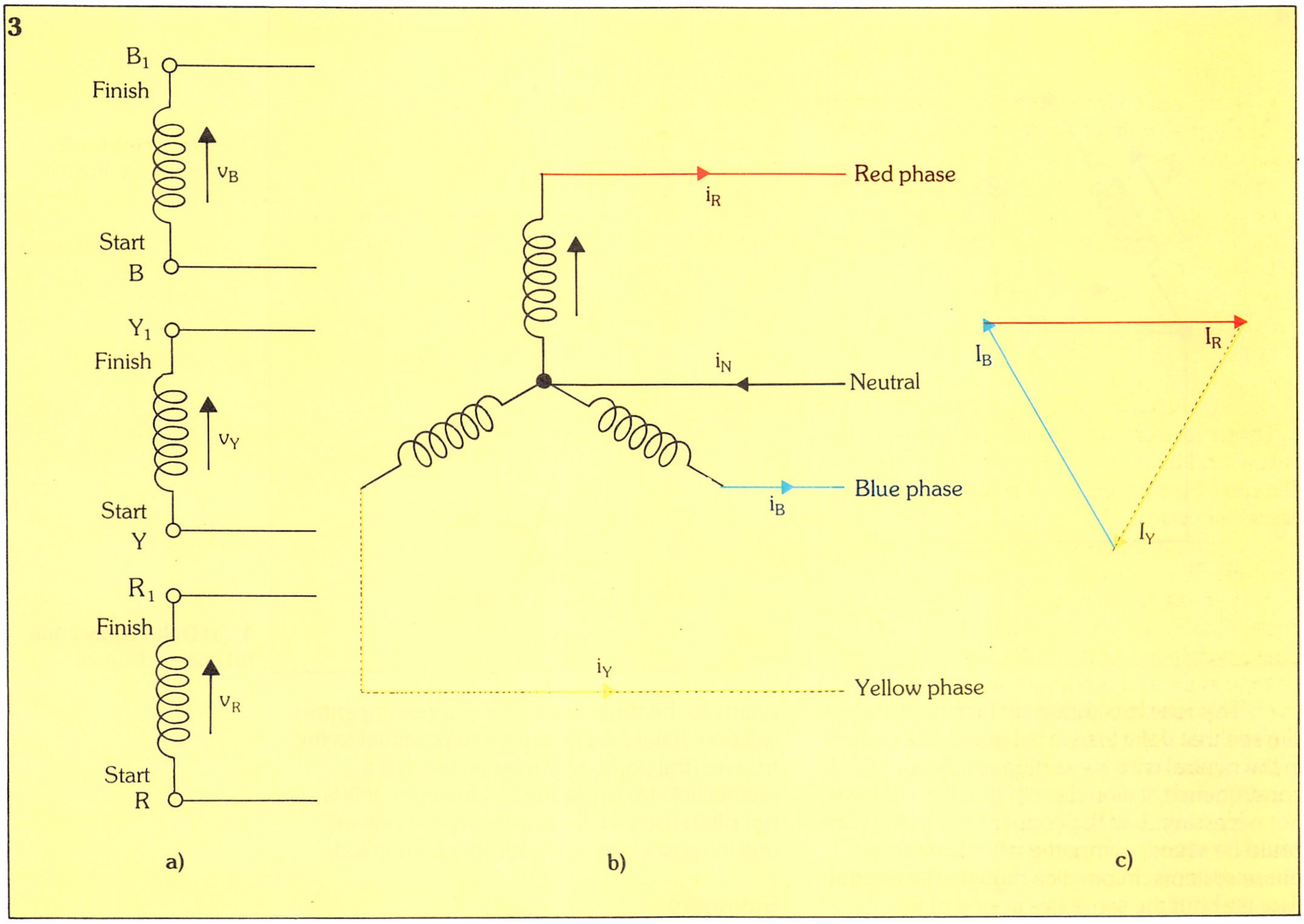
2. The EMFs generated in the coils RR_1, YY_1 and BB_1 make up a three-phase system.

1



2





3. (a) The three separate alternating supplies each with two leads; **(b)** star connection formed by joining the start ends of each stator winding; **(c)** phasor diagram.

way, and provides the ability to generate three times the power from this machine compared with a single phase alternator, without increasing its size.

We will now examine the way in which these three separate alternating supplies can be used. As the generator stands at the moment, there are three separate supplies, each of which has two leads. This is shown diagrammatically in *figure 3a*, where the three coils represent the separate coils in the alternator. Each of these can be used as a separate supply, providing power to its own set of consumers, but in the same way that a more powerful alternator can be made using the three-phase system, a more economical motor can be made by the same principle. Consequently, it is desirable to distribute the complete three-phase supply to those consumers with large motors, for example.

Star connection

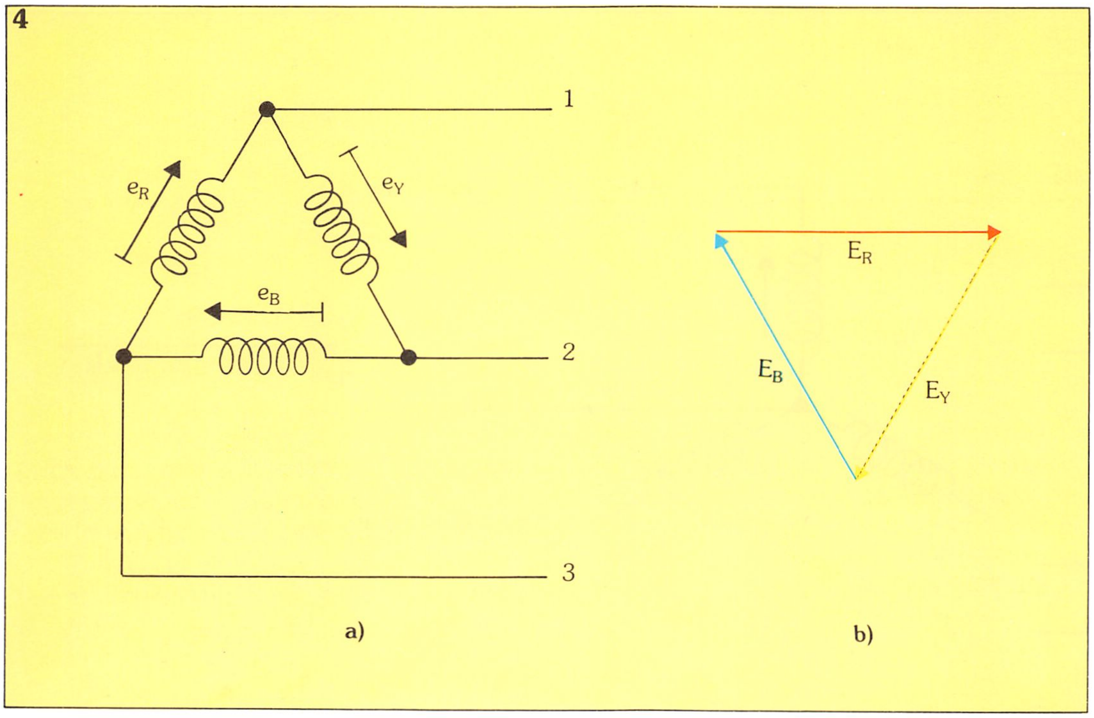
The three phases of the supply can be connected in one way, by joining the start ends of each stator winding. The power is then distributed to consumer(s) along four wires: the three, phase wires and the common neutral wire. This is known as a **star connection** and is shown in *figure 3b*. Ordinary domestic users only require a relatively small amount of power, so they are connected to neutral and only one of the phases. The electricity supply company, in fact, connects different domestic users in a street (often house by house) to different phases of the supply, so that the loading on each phase is as equal as possible. However, industrial consumers requiring a large amount of power are connected to all three phases.

Consider a number of consumers whose total load has been carefully balanced between the three phases. There will be equal currents flowing in the three-phase lines, and each will be in phase with its own voltage. The currents can thus be represented by a graph similar to that in *figure 2*. Each current is a sine wave and each phase current lags $2\pi/3$ behind the previous one. The current flowing in the neutral wire can be determined from:

$$i_N = i_R + i_Y + i_B$$

The phasor diagram in *figure 3c* represents this, and we can see that the total neutral current is zero.

4



4. (a) Delta connection (b) phasor diagram.

This is a very important fact, because we can see that if the load is balanced, the current in the neutral wire is exactly zero. As a consequence, it would seem that this wire was not necessary: half the copper used in the wires could be saved, compared with three single-phase systems. In practice though, the neutral wire is about the same size as one of the three-phase conductors, so only four wires are used instead of six.

Large electric motors driven by a three-phase supply naturally take a load which is completely balanced between the phases. The neutral wire is thus not needed and is often not supplied.

Delta connection

Figure 4a illustrates the delta (Δ) connection which can also be used to join the coils of a three-phase alternator. When this connection is used, care must be taken not to short circuit the windings into a loop. The phasor diagram of these three EMFs is redrawn in *figure 4b* – as you can see it is the same as the current phasor diagram for the star connection and this proves that the delta method is feasible.

You will no doubt have noticed that the delta method has no neutral wire and it is therefore not possible to connect unbalanced loads to the supplies.

When individual loads have to be connected between a phase line and neutral – and no neutral wire is supplied, an artificial star point is made. This is done by making a star of three equal resistors and connecting their points to the three lines. The star point (centre) will now have exactly the same potential as the true neutral point, and may be used as a connection for single loads. However, this is not often used as the resistors waste power, and in most cases a neutral wire *is* supplied.

Summary

Three-phase supplies allow electrical power to be generated and transmitted more economically than single-phase supplies. There is a smaller voltage drop and less power loss along the transmission lines between generator and consumer. Electric motors can also be started more easily and run more smoothly. □

27

DIGITAL ELECTRONICS

How digital systems function-2

Dedicated or multipurpose hardware

The third factor that we have to look at in the design of digital circuits is the choice between using dedicated hardware and multipurpose hardware, to carry out a specific function. In other words, do we use a special unit by itself, dedicated to just one job – or do we use a unit that can do this job and others as well?

Again, the calculator can provide us with examples of each choice. Consider the addition function: the program counter contains a dedicated adder, which is limited to one single specific job – that of adding one to the address register to obtain the next instruction; the adder-subractor, on the other hand, is a multipurpose unit that can perform the addition function. It is multipurpose not only because it can perform addition, subtraction and comparison, but also because it works out where the numbers to be added come from and where the sums go.

We could even make a calculator that used the adder-subtractor to increment the program counter – thus using a multipurpose rather than a dedicated adder. However, that would make the programming very clumsy, clutter up the microprogram memory with extra instructions for adding to addresses and greatly increase the time required for the calculator to do anything.

So, as you can see, the choice between dedicated and multipurpose depends on the circumstances. Generally speaking, a very simple function that requires very little hardware is best performed in a dedicated manner. The program counter's adder, for example, is a relatively insignificant part of the calculator chip. However, if a sizeable piece of hardware is needed to perform a function then it will be much more economically used if it is shared with the other applications in the system. This is the rationale behind using a single unit for adding, subtracting and comparison. With only a little extra hardware and programming, a subsystem can be made which performs all three jobs that make up the adder-subtractor's function.

Right: Caere hand-held optical character reader. This model is user programmable with standard parallel TTL and RS232C interfaces. (Photo: Cognitronics Ltd).

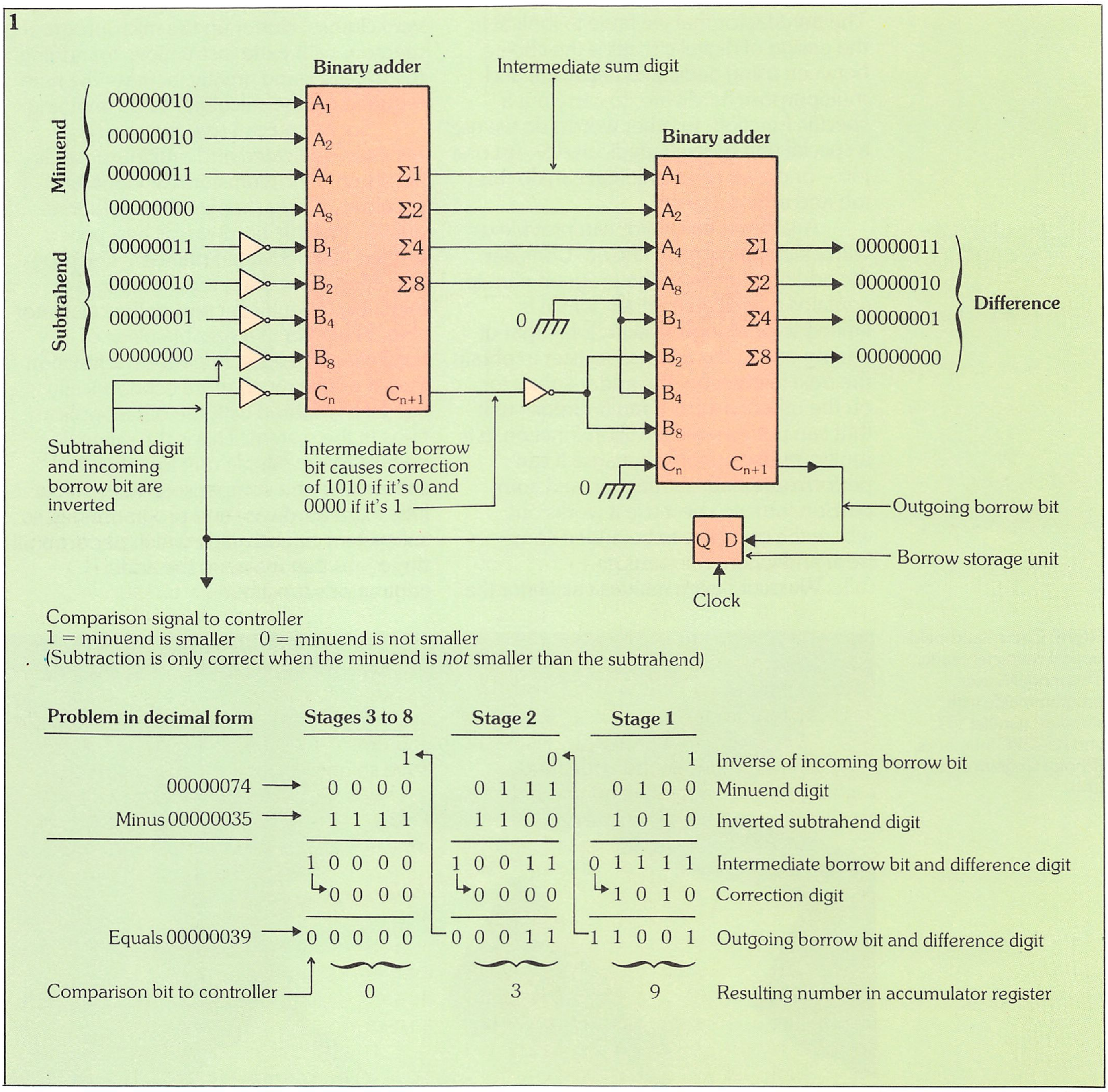
Further examples of system design

We have now looked at the three main design decisions that need to be made when designing digital systems: parallel or serial processing; hard-wired or variable programmed control; and dedicated or multipurpose hardware. The calculator will again provide us with more examples of these choices.

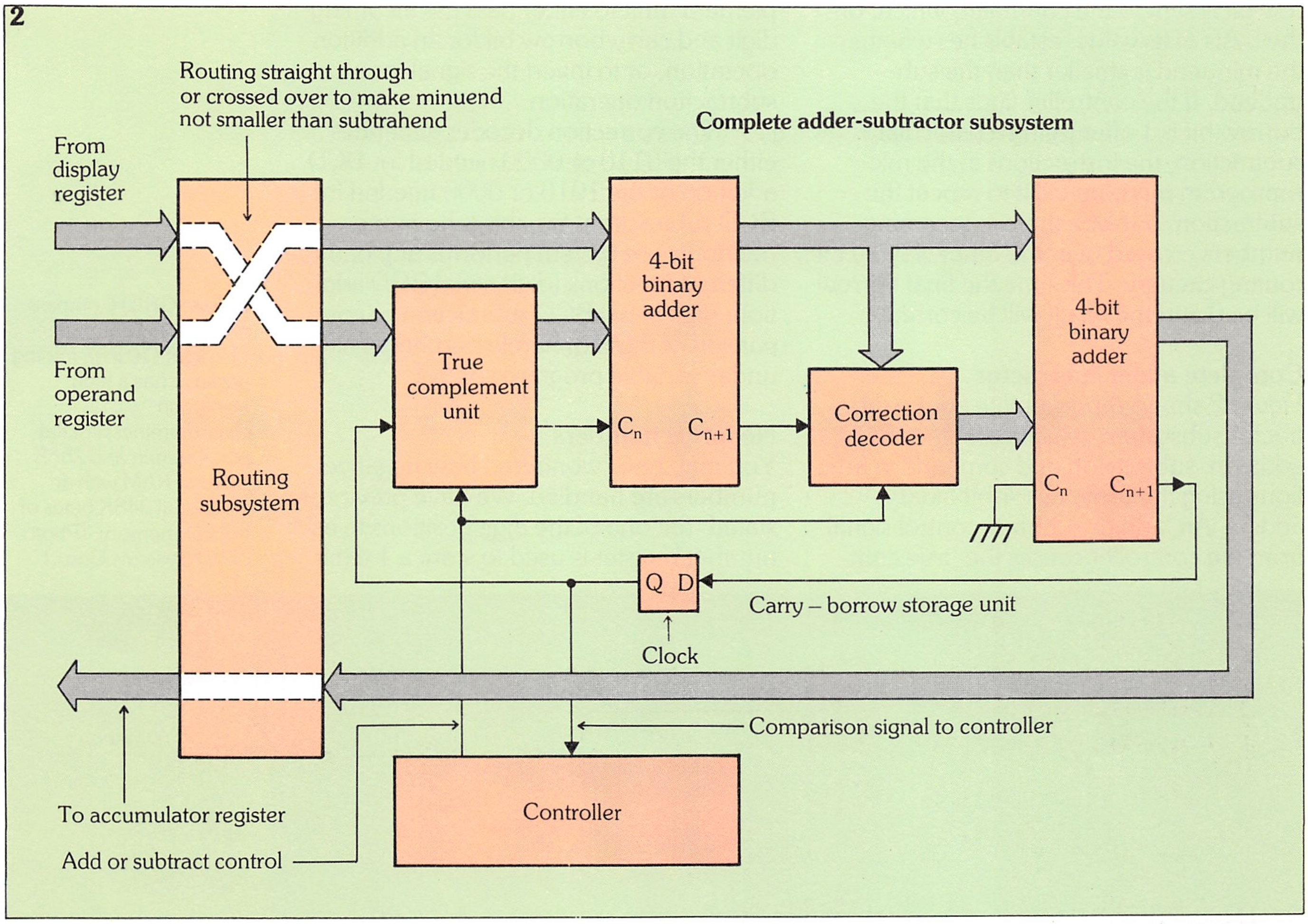
BCD subtraction using one's complement

Figure 1 shows one of the main methods used to subtract BCD numbers, and this uses the one's complement of the subtrahend. (Remember, the subtrahend is the number subtracted, while the minuend is the number subtracted from.) To make a number into its one's complement, all the bits are inverted – the ones are changed to zeros and vice versa. This changes the sign of the number and enables it to be

1. Possible design for a bit parallel, digit serial one's complement BCD subtractor, showing the subtraction of 35 from 74.



2



2. Possible design for a complete adder-subtractor subsystem.

subtracted, by adding its one's complement in a certain way.

As *figure 1* shows, the 4-bit subtrahend digit and the incoming carry (now called **borrow**) are first inverted. The resultant bits are added to the minuend digit by a binary adder. The resulting **intermediate sum** digit goes to a second adder. There, it's added to 1010 (ten) if the first addition produced a carry (borrow) of zero. If this first (intermediate) carry (borrow) bit was 1, the second adder adds nothing (0000). When the minuend digit is smaller than the subtrahend digit, an outgoing borrow of 1 is produced and held in the carry/borrow storage unit like the carry bit during addition.

The example shown in *figure 1* subtracts 35 from 74. The various bits are shown in the order that they are clocked in and out of the subtractor. The details of the process are listed at the bottom. Here, the first and second additions of each stage are shown, one above the other, and you should work through the process yourself from right to left. There are eight stages, for the eight digit baskets in the registers, and stages three to eight are identical. The original incoming borrow bit is 0.

This subtractor only works correctly when the minuend is not smaller than the subtrahend, i.e. when the minuend is greater than or equal to the subtrahend. To understand why, look back to our example and notice that after all eight pairs of digits have shifted through the subtractor, the last borrow bit left in the storage unit is 0. This signifies that the whole minuend is not smaller than the subtrahend. If we had subtracted 74 from 35 instead, we would have ended up with an answer of 99999961, with a final borrow of 1. This borrow tells us that the minuend is smaller than the subtrahend, and that we have to do something different to obtain the right answer.

The subtractor can also perform the function of a **comparator**. The final bor-

row bit is sent to the controller, which, on the basis of its value, establishes whether the minuend is smaller than the subtrahend. If the controller finds that the borrow bit is 1 after trying a particular subtraction, the instructions in the microprogram memory tell it to repeat the subtraction, but with the two incoming numbers crossed over the other way by the routing circuitry. This time the final borrow will be 0 and the result will be correct.

Complete adder subtractor

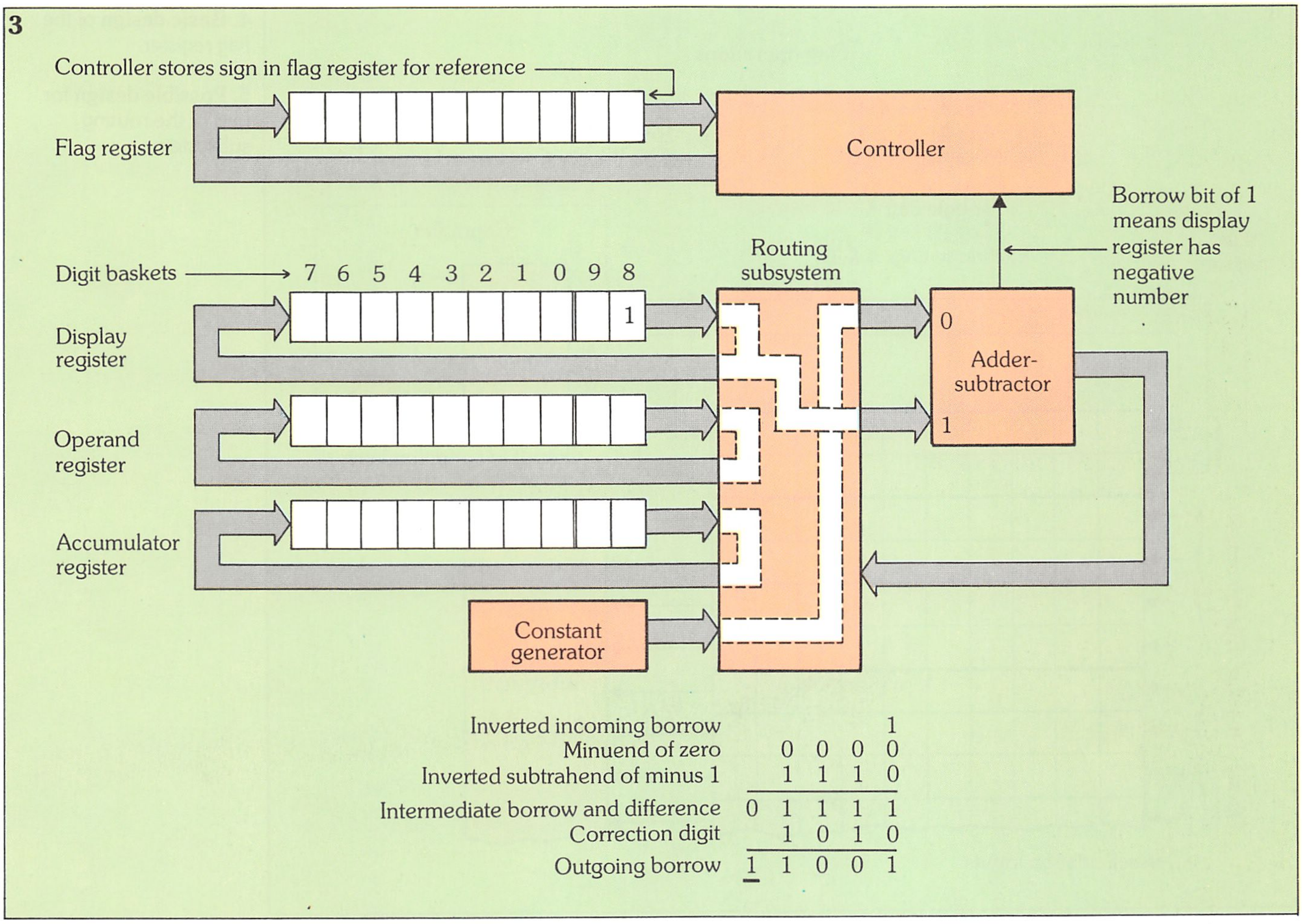
Figure 2 shows the complete adder-subtractor subsystem, which combines BCD addition, subtraction and comparison functions using the same two 4-bit binary adders. An 'add or subtract' control signal from the controller causes the 'true complement' unit to either pass the incoming digit and carry/borrow bit for an addition operation, or to invert the signal for a subtraction operation.

The correction decoder generates either the 0110 or 0000 needed for BCD addition or the 1010 or 0000 needed for BCD subtraction. So, this is how one multipurpose system performs any of three different functions (digit-serial BCD addition, digit-serial BCD subtraction or comparison of digits or whole numbers), all under variable program control.

Negative numbers

You may have wondered how negative numbers are handled. We have previously stated that one of the digit positions in each number register is used to store a 1 if the

Below: GRID Compass professional computer. This light (10 lbs), folding package has a high resolution, electroluminescent flat-panel screen and 256K bytes of RAM with an additional 348K bytes of bubble memory. (Photo: GRID Systems Corp.)



3. The controller notes the sign of the stored numbers by comparing the sign digit with zero from the constant generator.

number is negative – but how does the calculator know what sign to give to a number when it performs an arithmetic operation?

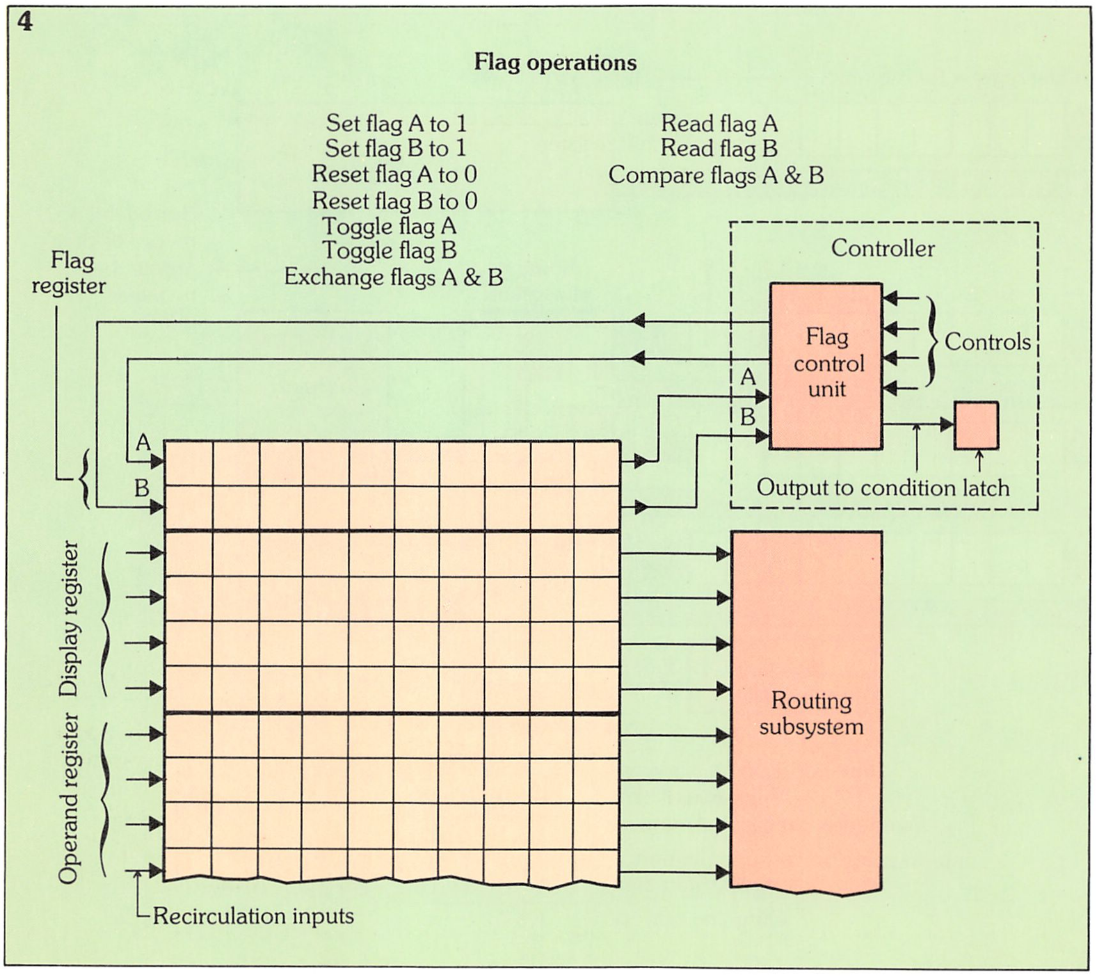
The answer to this question is that the controller follows a sequence of instructions to examine the signs of the two original numbers in the display and operand registers. The controller then uses some simple logic to work out what the sign of the result should be. In the case of adding and subtracting, this also requires establishing which of the two numbers is the larger. The controller examines signs in the same way that it compares the size of numbers, by the comparison function.

For example, to check the sign of the number in the display register in *figure 3*, the controller waits until the sign digits (eighth basket position) are at the register outputs. Then it subtracts the display register sign bit from zero (which is supplied by the 'constant generator' subsystem). A resulting borrow of 1 informs the controller that the display register sign is negative.

The controller then stores this sign in the flag register and notes which number is the largest. It then follows a set of instructions that leads it to work out what the sign of the result in the accumulator register should be and routes a 1 or 0 from the constant generator to the sign-digit position in the accumulator register.

So, how does the flag register work? *Figure 4* shows a basic design that is compact and convenient. The flag register consists of two 10-bit recirculating registers, A and B. The flag register and all three number registers are made as a single mass memory unit, containing fourteen 10-bit shift registers, all shifting in step together. This means that one basket at a time is accessible by the controller.

Each basket contains two flag bits that can be read or written as the basket passes through the controller. The flag control unit in the controller subsystem (*figure 4*) handles



4. Basic design of the flag register.

5. Possible design for part of the routing subsystem.

the reading, writing and recirculation of the flags. A possible list of different flag operations is shown, and these can be performed on the two flag bits in one basket by sending the appropriate signals to the control unit. These signals would be sent by an instruction word from the microprogram memory. The bit resulting from a **flag-test operation** (reading or comparing flags) is held in a 1-bit storage unit in the controller, called the **condition latch**. A later instruction instructs the controller what to do on the basis of the latch's logic state.

We've mentioned the routing subsystem so often that we ought to see how it works. This consists mainly of data selectors, which can be arranged as shown in *figure 5*. Here, for simplicity, only the part of the subsystem relating to the display register and one input to the adder-subtractor is shown.

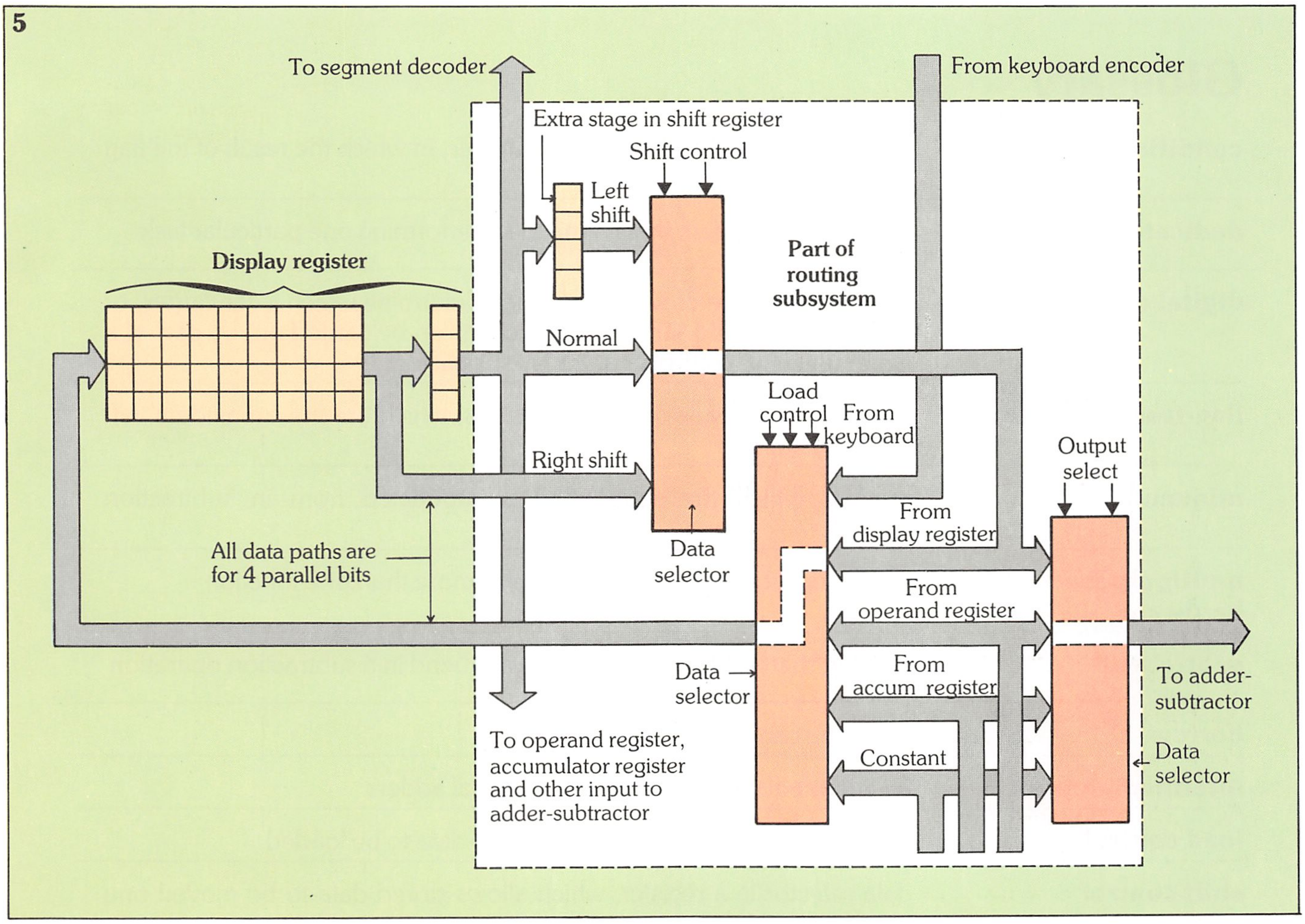
Each data selector has a number of 4-bit inputs and one 4-bit output – one for each decimal digit. One 4-bit input digit to each data selector is selected by the controller for transmission to the output. As an example, the broken lines in *figure 5* show the paths selected by the controller to recirculate digits in the display register and also route them to one input of the adder-subtractor.

Figure 5 also includes a new feature – a method of shifting stored digits to the right or left among the ten baskets in each number register. Remember, before a new digit can be entered from the keyboard into the far right end of the display register, the stored digits have to be shifted one to the left. And to 'line up the decimal points' before adding or subtracting, we may have to shift numbers to the left or right.

This is carried out by controlling the **shift control** data selector at the left in *figure 5*, while the **load control** data selector is set to recirculate data. During one full recirculation of the register, the shift control causes either one shift to the

Right: the MITEL SX-2000 Integrated Communications System, shown with the Superset 7 operator's console. This digital switch can operate voice, data and image simultaneously. (Photo: MITEL).

5



left (by selecting the upper input), no shift at all (by selecting the middle input), or one shift to the right (by selecting the lower input).

The left shift path simply adds one extra stage to the shift register, thus delaying the recirculating digits one step behind their former positions. The right shift path short circuits the recirculation path by one stage, by taking data from the next to the last stage of the display register. This advances the digits one position ahead of where they were before.

Conclusions

We have now covered the various types of applications and the problems that the engineer must solve before coming to grips with the detailed design of a digital system.

The fundamental concepts have now been covered. The controlling power behind most sophisticated digital circuits and computers – the microprocessor – will be explored in a forthcoming series.

Glossary

condition latch	1-bit register in the calculator's controller, in which the result of the flag test operation is stored
dedicated hardware	system or subsystem that is limited to performing one particular task
digital comparator	compares the size of two numbers to determine which is the larger, or if they are both the same size. Function can be carried out by part of a larger unit (e.g. controller or adder-subtractor)
flag-test operation	operation that reads or compares flags, which are the 'reminders' left by the system
minuend	number which the subtrahend is subtracted from in subtraction operation
multipurpose hardware	system or subsystem that can perform more than one function
subtrahend	number that is subtracted from the minuend in a subtraction operation
borrow	incoming carry bits to a digital system
intermediate sum	an intermediate digit between two digital adders
load control	data selector in a register, which allows data to be loaded
shift control	data selector in a register, which allows stored data to be moved one space right or left